NUREG/CR-6150
EGG-2720
Volume III

SCDAP/RELAP5/MOD3.1 Code Manual

Volume III: SCDAP/RELAP5 Users Guide and Input Manual

Contributing Authors:
C. M. Allison
G. A. Berna
E. W. Coryell
K. L. Davis
D. T. Hagrman
J. K. Hohorst
R. R. Schultz
L. J. Siefken

Editors:
E. W. Coryell
E. C. Johnsen

Idaho National Engineering Laboratory
EG&G Idaho, Inc.
Idaho Falls, Idaho 83415

Prepared for the
Division of Systems Research
Office of Nuclear Regulatory Research
U. S. Nuclear Regulatory Commission
Washington, D.C. 20555
Under DOE Contract No. DE-AC07-761D01570
FIN W6095

ABSTRACT

The SCDAP/RELAP5 code has been developed for best estimate transient simulation of light water reactor coolant systems during a severe accident. The code models the coupled behavior of the reactor coolant system, core, fission product released during a severe accident transient as well as large and small break loss of coolant accidents, operational transients such as anticipated transient without SCRAM, loss of offsite power, loss of feedwater, and loss of flow. A generic modeling approach is used that permits as much of a particular system to be modeled as necessary. Control system and secondary system components are included to permit modeling of plant controls, turbines, condensers, and secondary feedwater conditioning systems.

This volume provides guidelines to code users based upon lessons learned during the developmental assessment process. A description of problem control and the installation process is included. Appendix A contains the description of the input requirements.

FIN W6095-SCDAP/RELAP5 CODE IMPROVEMENT AND ASSESSMENT

EXECUTIVE SUMMARY

The specific features of SCDAP/RELAP5/MOD3.1 are described in this five volume set of manuals covering the theory, use, and assessment of the code for severe applications. This set replaces the SCDAP/RELAP5/MOD3 Code Manual, NUREG/CR-5273.

The SCDAP/RELAP5 computer code is designed to describe the overall reactor coolant system (RCS) thermal-hydraulic response, core damage progression, and in combination with VICTORIA[a], fission product release and transport during severe accidents. The code is being developed at the Idaho National Engineering Laboratory (INEL) under the primary sponsorship of the Office of Nuclear Regulatory Research of the U.S. Nuclear Regulatory Commission (NRC).

The code is the result of merging the RELAP5/MOD3[b] and SCDAP models. The RELAP5 models calculate the overall RCS thermal hydraulics, control system interactions, reactor kinetics, and the transport of noncondensable gases. Although previous versions of the code have included the analysis of fission product transport and deposition behavior using models derived from TRAP-MELT, this capability is being replaced through a data link to the detailed fission product code, VICTORIA, as a result of an effort to reduce duplicative model development and assessment.

The SCDAP code models the core behavior during a severe accident. Treatment of the core includes fuel rod heatup, ballooning and rupture, fission product release, rapid oxidation, zircaloy melting, UO_2 dissolution, ZrO_2 breach, flow and freezing of molten fuel and cladding, and debris formation and behavior. The code also models control rod and flow shroud behavior.

The development of the current version of the code was started in the spring of 1992. This version contains a number of significant improvements to the SCDAP models since the last versions of the code, SCDAP/RELAP5/MOD2.5 and SCDAP/RELAP5/MOD3[7af], were released. These improvements include the addition of several new models to describe the earlier phases of a severe accident, changes in the late phase models to provide more “physically intuitive” behavior for full plant calculations, and changes to improve the overall reliability and usability of the code. The improvements in the early phase models include the addition of models to treat (a) the effects of grid spacers including the effects of Inconel spacer grid-zircaloy cladding material interactions, (b) BWR B_4C control blade-zircaloy channel box material interactions, and (c) accelerated heating, melting, and hydrogen generation during the reflood of damaged fuel rods. An extension to the molten pool models to treat the sporadic growth of the boundaries of the molten pool into adjacent regions of relatively intact assemblies or rubble debris beds is the most significant change to the late phase models. Improvements in overall reliability and usability of the code for plant calculations include changes in the overall code numerics to reduce the likelihood of numerical instabilities or code failures and changes in the codes input/output processors. The most noticeable of these for the code users is the conversion of the SCDAP input to a form more compatible with the RELAP5 style. In addition to the these modeling and coding changes, SCDAP/RELAP5/MOD3.1 has also been subjected to (a) an intensive effort of verification testing to identify and resolve outstanding code errors and

---

a. T. Heames et al., *VICTORIA: A Mechanistic Model of Radionuclide Behavior in the Reactor Coolant System Under Sever Accident Conditions*, NUREG/CR-5545, SAND90-0756, Rev. 1, December 1992.

b. C. M. Allison, C. S. Miller, N. L. Wade (Eds.) *RELAP5/MOD3 Code Manual*, Volumes I through IV, NUREG/CR-5535, EGG-2596, June 1990.

(b) a systematic assessment of the code to quantify the uncertainties in the predicted results.

The RELAP5 code is based on a nonhomogeneous and nonequilibrium model for the two-phase system that is solved by a fast, partially implicit numerical scheme to permit economical calculation of system transients. The objective the RELAP5 development effort from the outset was to produce a code that includes important first order effects necessary for accurate prediction of system transients but is sufficiently simple and cost effective such that parametric or sensitivity studies are possible. The development of SCDAP/RELAP5 has this same focus.

The code includes many generic component models from which general systems can be simulated. The component models include fuel rods, control rods, pumps, valves, pipes, heat structures, reactor point kinetics, electric heaters, jet pumps, turbines, separators, accumulators, and control system components. In addition, special process models are included for effects such as form loss, flow at an abrupt area change, branching, choked flow, boron tracking, and noncondensable gas transport.

This volume, Volume III, gives detailed descriptions of the input preparation and execution procedures. It also provides code installation procedures, as well as general guidelines on code applications.

ACKNOWLEDGMENTS

Development of a complex computer code such as SCDAP/RELAP5 is the result of a team effort. Acknowledgments are made to those who made significant contributions to the earlier versions of SCDAP/RELAP5 in particular for the contributions of G. H. Beers, E. R. Carlson, and T. M. Howe. Acknowledgment is also made of E. C. Johnsen for her work in code configuration control, to B. D. Reagan for her support in preparing figures and correcting equations, and to L. G. Price and N. Wade for technical editing support. The authors also acknowledge the RELAP5 development team, specifically R. J. Wagner, R. A. Riemke, and C. S. Miller for their contributions to SCDAP/RELAP5.

The SCDAP/RELAP5 Program is indebted to the technical monitors responsible for directing the overall program: Dr. Y. Chen of the U. S. Nuclear Regulatory Commission and Mr. W. H. Rettig of the Department of Energy Idaho Operations Office. Finally, acknowledgment is made of those many code users who have been very helpful in stimulating correction of code deficiencies with special appreciation to Dr. T. J. Haste and L. Nilsson for the many hours of technical review they provided.

CONTENTS

SCDAP/RELAP5/MOD3.1[a] Code Manual Volume III: SCDAP/RELAP5 User's Guide and Input Manual

1. INTRODUCTION

The SCDAP/RELAP5 computer code is designed to describe the overall reactor coolant system (RCS) thermal-hydraulic response, core damage progression, and, in combination with VICTORIA,[1] fission product release and transport during severe accidents. The code is being developed at the Idaho National Engineering Laboratory (INEL) under the primary sponsorship of the Office of Nuclear Regulatory Research of the U.S. Nuclear Regulatory Commission (NRC).

1.1 General Code Capabilities

The code is the result of merging the RELAP5/MOD3[2] and SCDAP[3] models. The RELAP5 models calculate the overall RCS thermal hydraulics, control system interactions, reactor kinetics, and transport of noncondensable gases. Although previous versions of the code have included the analysis of fission product transport and deposition behavior using models derived from TRAP-MELT, this capability is being replaced through a data link to the detailed fission product code, VICTORIA, as a result of an effort to reduce duplicative model development and assessment. The SCDAP models calculate the damage progression in the core structures and the formation, heatup, and melting of debris.

SCDAP/RELAP5 is capable of modeling a wide range of system configurations from single pipes to different experimental facilities to full-scale reactor systems. The configurations can be modeled using an arbitrary number of fluid control volumes and connecting junctions, heat structures, core components, and system components. Flow areas, volumes, and flow resistances can vary with time through either user control or models that describe the changes in geometry associated with damage in the core. System structures can be modeled with RELAP5 heat structures, SCDAP core components, or SCDAP debris models. The RELAP5 heat structures are one-dimensional models with slab, cylindrical, or spherical geometries. The SCDAP core components include representative light water reactor (LWR) fuel rods, silver-indium-cadmium (Ag-In-Cd) and B_4C control rods and/or blades, electrically heated fuel rod simulators, and general structures. A two-dimensional, finite element model based upon the COUPLE4 code may be used to calculate the heatup of debris and/or surrounding structures. This model takes into account the decay heat and internal energy of newly fallen or formed debris and then calculates the transport by conduction of this heat in the radial and axial directions to the wall structures and water

a. The MOD3.1 designates a generic version of the code. Where needed, specific developmental version identification will be included with the code name. For example, SCDAP/RELAP5/MOD3[8x] would specify developmental version 8x.

surrounding the debris. Perhaps the most important use of this model is to calculate the heatup of the vessel wall so that the time at which the vessel may rupture can be determined. Other system components available to the user include pumps, valves, electric heaters, jet pumps, turbines, separators, and accumulators. Models to describe selected processes, such as reactor kinetics, control system response, and tracking noncondensable gases, can be invoked through user control.

The development of the current version of the code was started in the spring of 1992. This version contains a number of significant improvements since the last versions of the code, SCDAP/RELAP5/MOD2.5 and SCDAP/RELAP5/MOD3[7af], were released. These improvements include the addition of several new models to describe the earlier phases of a severe accident, changes in the late phase models to provide more "physically intuitive" behavior for full plant calculations, and changes to improve the overall reliability and usability of the code. The improvements in the early phase models include the addition of models to treat (a) the effects of grid spacers including the effects of Inconel spacer grid-zircaloy cladding material interactions, (b) BWR B_4C control blade-zircaloy channel box material interactions, and (c) accelerated heating, melting, and hydrogen generation during the reflood of damaged fuel rods. An extension to the molten pool models to treat the sporadic growth of the boundaries of the molten pool into adjacent regions of relatively intact assemblies or rubble debris beds is the most significant change to the late phase models. Improvements in overall reliability and usability of the code for plant calculations include changes in the overall code numerics to reduce the likelihood of numerical instabilities or code failures and changes in the codes input/output processors. The most noticeable of these for the code users is the conversion of the SCDAP input to a form more compatible with the RELAP5 style. In addition to these modeling and coding changes, SCDAP/RELAP5/MOD3.1 has also been subjected to (a) an intensive effort of verification testing to identify and resolve outstanding code errors and (b) a systematic assessment of the code to quantify uncertainties in the predicted results.

1.2 Relationship to Other NRC-Sponsored Software

SCDAP/RELAP5 and RELAP5 are developed in parallel and share a common configuration. Both codes share a common source deck. Separate codes are formed only prior to compilation, so changes made to the source deck are automatically reflected in both codes.

The development and application of the code is also related to several other NRC-sponsored software packages. Theoretical work associated with the development of PARAGRASS-VFP,[5] has resulted in model improvements for fission product release. A data link to the VICTORIA code will allow for the detailed treatment of phenomena such as fission product and aerosol transport, deposition, and resuspension. A link with PATRAN[6] and ABAQUS[7] provides the user with the means to calculate the details of lower head failure. Animated plant response displays are possible through links to the Nuclear Plant Analyzer (NPA)[8] display software, which gives the user an efficient way of analyzing the large amount of data generated. Detailed plant simulations from accident initiation through release of fission products to the atmosphere are made available through links to the CONTAIN[9] containment response and CRAC2[10] or MACCS[11] atmospheric dispersion consequence codes.

1.3 Quality Assurance

SCDAP/RELAP5 is maintained under a strict code configuration system that provides a historical record of the changes made to the code. Changes are made using an update processor that allows separate identification of improvements made to each successive version of the code. Modifications and improvements to the coding are reviewed and checked as part of a formal quality program for software. In addition, the theory and implementation of code improvements are validated through assessment calculations that compare the code-predicted results to idealized test cases or experimental results.

1.4 Organization of the SCDAP/RELAP5 Manuals

The specific features of SCDAP/RELAP5/MOD3.1 are described in a five-volume set of manuals covering the theory, use, and assessment of the code for severe applications. Although Volume I describes (a) the overall code architecture, (b) interfaces between the RELAP5 system models, and (c) any system models unique to SCDAP/RELAP5, the code user is referred to the companion set of six volumes which describe the RELAP5[2] system thermal hydraulics and associated models.

Volume I presents a description of SCDAP/RELAP5/MOD3.1-specific thermal-hydraulic models (relative to RELAP5/MOD3), and interfaces between the thermal-hydraulic models and damage progression models.

Volume II contains detailed descriptions of the severe accident models and correlations. It provides the user with the underlying assumptions and simplifications used to generate and implement the basic equations into the code, so an intelligent assessment of the applicability and accuracy of the resulting calculation can be made.

Volume III provides the user's guide and code input for the severe accident modeling. SCDAP input was recently changed to be consistent with the free-form input used by RELAP5. User guidelines are produced specifically for the severe accident code. The user should also refer to the RELAP5/MOD3 Code Manual Volume V: User Guidelines for a complete set of guidelines.

Volume IV describes the material property library, MATPRO. It contains descriptions of the material property subroutines available for severe accident analysis.

Volume V documents the assessments of SCDAP/RELAP5/MOD3.1 created in early 1993. It includes nodalization sensitivity studies and time-step sensitivity studies, assessments using standard PWR and BWR plant models, and assessments using code-to-data comparisons.

1.5 Organization of Volume III

The purpose of this volume is to help educate the code user by documenting the modeling experience that has been accumulated from developmental assessment and application of the RELAP5 and SCDAP codes. This information will include a blend of the model developer recommendations with respect to how

the model was intended to be applied and the application experience that indicates what has been found to work or not to work. Where possible, definite recommendations of approaches known to work are made and approaches known not to work are pointed out as pitfalls to avoid.

The objective of the user's guide is to reduce the uncertainty associated with severe accident modeling of light water reactor (LWR) coolant systems. However, we do not imply that uncertainty can be eliminated or even quantified in all cases, since the range of possible system configurations and transients that could occur is large and constantly evolving. Hence, the effects of nodalization, time step selection, and modeling approach are not completely quantified. As the assessment proceeds, there will be a continual need to update the user guidelines document to reflect the current state of modeling knowledge.

Section 2 describes the type of core structures that can be modeled with SCDAP/RELAP5. Section 3 provides user guidelines, and section 4 describes problem control and output editing. Section 5 discusses the installation process. Appendix A documents the input requirements for SCDAP/RELAP5.

2. CORE STRUCTURES

The core structures represent those portions of the reactor core which are solid at the beginning of the analysis. This may include fuel rods, control rods, flow shrouds, simulator rods, or channel boxes.

2.1 Fuel Rod

The fuel rod behavior model calculates the thermal, mechanical, and chemical response of fuel rods during severe accidents. The fuel rod behavior models consider nuclear heat generation, temperature distribution, zircaloy cladding oxidation, fuel deformation, liquefaction, and fission product release. Nuclear heat generation, in combination with the heat generation of cladding oxidation, determines the fuel rod temperature. The rod temperature is computed by a two-dimensional finite difference scheme. The oxidation heat of zircaloy is the dominant heat source after temperatures reach 1,500 K. Cladding deformation is based on mechanical models developed for FRAP-T6[12] and FRAPCON-2.[13] The model considers both axisymmetric cladding collapse or ballooning and asymmetric localized ballooning. The melt, flow, and refreezing of liquefied U-O-Zr is also considered. The liquid material is assumed to flow as an axisymmetric slug depositing both heat and a frozen crust upon the underlying ZrO_2 layer. The release of inert gases (krypton, xenon, helium) and volatile fission product (cesium, iodine) is modeled using the PARAGRASS[5] model.

2.2 Ag-In-Cd Control Rod

Control rod temperatures are computed using the same heat conduction model as the fuel rods. User specified nuclear heating, chemical heating caused by oxidation of the zircaloy guide tube and stainless steel cladding, and convective and radiative heat transfer from the coolant and adjacent fuel rods are considered. The melting and relocation of control rod materials are described in the following manner. If the stainless steel is below its melting temperature, no relocation of molten Ag-In-Cd occurs. If the guide tube melts, or is breached, molten absorber moves through the breach in the zircaloy guide tube and moves as a film on the outside of the guide tube. Unlike the flow of molten Zr-U-O for fuel rods, the momentum and energy equations are not solved to describe the freezing of the molten Ag-In-Cd; rather, the material freezes when it reaches a lower elevation where the guide tube temperature is 200 K less than the solid temperatures of Ag-In-Cd. For subsequent heatup and melting of stainless steel and zircaloy, the molten material relocates internally downward within the oxidized ZrO_2 on the guide tube, filling up the voids formed by the relocation of molten Ag-In-Cd. The molten mixture of stainless steel and zircaloy will remain contained within the ZrO_2 shell until the ZrO_2 is either melted, allowing the molten mixture to flow downward in the flow channel until it freezes, or is shattered upon reflood.

2.3 Flow Shroud

The structures internal to the core other than fuel and control rods can be modeled using the basic heat conduction equation. Heat generation can be user specified and oxidation related. The structures can be defined by multiple layers of materials, with the oxidation and relocation of exterior layers caused by melting considered. Zircaloy layers are oxidized using the same kinetics as described for fuel rods. The molten zircaloy relocates downward to a region where the structural surface temperature can also be

modeled; however, oxidation rate equations must be user specified and no material relocation or loss of geometry can be considered. Both melting and non-melting models can be used for the structures outside the core as well, since the same material limitations apply.

2.4 Simulator Rod

The simulator rod is used in out of pile experiments to simulate the behavior of fuel rods during a severe accident scenario. The simulator rod is heated electrically by tungsten wire at the center. The simulator rod behavior model calculates the thermal, mechanical, and chemical response of simulator rods during severe accidents. The model considers electric heat generation, temperature distribution, zircaloy cladding oxidation, and fuel deformation and liquefaction. Electric heat generation, in combination with the heat generation of cladding oxidation, determines the fuel rod temperature. The rod temperature is computed by a two-dimensional finite difference scheme. Cladding deformation is based on mechanical models developed for FRAP-T6 and FRAPCON-2. The melt, flow, and refreezing of liquefied U-O-Zr are also considered.

2.5 BWR Control Blade/Channel Box Model

Analyses of the DF-4 and CORA experiments have shown that the effects of B_4C/stainless steel interactions, as well as stainless steel/Zircaloy interactions, must be included to predict control blade relocation accurately. Melting of a control blade begins at the inner surfaces of the absorber rodlets where stainless steel reacts with B_4C. The absorber rodlets fail at a temperature that is lower than the melting temperature of pure stainless steel. Stainless steel from the control blade then relocates downward and forms a blockage between the control blade and channel box, where it reacts with the Zircaloy. The Zircaloy channel box adjacent to the stainless steel blockage fails (enters into the formation of a eutectic mixture) at a temperature that is much lower than the melting of pure Zircaloy.

2.6 B_4C Control Rod Model

The B_4C control rod model remains in the code for several reasons. First the new control blade / channel box model does not yet model relocation of molten control rod material through the channel box wall and into the fuel rod bundle. Therefore, although this model is not as sophisticated as the combined control blade / channel box model described in Section 2.5, this model must still be used to extend analyses past the time of control rod relocation. Secondly, some reactors do make use of cylindrical B_4C control rods.

Control rod temperatures are computed using the same heat conduction model as the fuel rods. User specified nuclear heating, chemical heating caused by oxidation of the zircaloy guide tube and stainless steel cladding, and convective and radiative heat transfer from the coolant and adjacent fuel rods are considered. The melting and relocation of control rod materials are described in the following manner. If the stainless steel is below its melting temperature, no relocation of molten Ag-In-Cd occurs. If the guide tube melts, or is breached, molten absorber moves through the breach in the zircaloy guide tube and moves as a film on the outside of the guide tube. Unlike the flow of molten Zr-U-O for fuel rods, the momentum and energy equations are not solved to describe the freezing of the molten Ag-In-Cd; rather, the material freezes when it reaches a lower elevation where the guide tube temperature is 200 K less than the solid

temperatures of Ag-In-Cd. For subsequent heatup and melting of stainless steel and zircaloy, the molten material relocates internally downward within the oxidized ZrO2 on the guide tube, filling up the voids formed by the relocation of molten B_4C. The molten mixture of stainless steel and zircaloy will remain contained within the ZrO2 shell until the ZrO2 is either melted, allowing the molten mixture to flow downward in the flow channel until it freezes, or is shattered upon reflood.

3. SCDAP/RELAP5 USER GUIDELINES

The objective of the user guidelines is to reduce the uncertainty associated with severe accident modeling of light water reactor (LWR) coolant systems. However, we do not imply that uncertainty can be eliminated or even quantified in all cases, since the range of possible system configurations and transients that could occur is large and constantly evolving. As experience with the code is gained, there will be a continual need to update the user guidelines document to reflect the current state of modeling knowledge.

3.1 SCDAP/RELAP5 Card Number Input

The SCDAP/RELAP5 input structure has traditionally comprised three different styles--RELAP5 card number, SCDAP unformatted, and COUPLE fixed format. While the SCDAP unformatted input was free-form, it provided no capability for input checking or error recovery during the input process. The COUPLE input scheme required that the input be right-justified within a specified range of columns. This input structure made creation of severe core accident analysis input decks time-consuming, frustrating, and unreliable due to extremely primitive levels of input checking. The input had virtually no error detection, and bad input often ended in floating point exceptions and I/O errors. Resolving input errors often required knowledge of the code structure, use of debugging tools, and use of code debug printout. Since RELAP5-style card-number input provided significantly greater flexibility in input checking, and since users are already familiar with this style of input, all SCDAP/RELAP5 input has been converted to use RELAP5-style card-number input.

Input for SCDAP/RELAP5 is processed on three levels, (a) input echo; (b) individual card, or R level processing; and (c) initialization, or I level processing. This input philosophy provides the maximum diagnostic information for each input submittal. During input echoing, the input deck is echoed to the output file; and cards with the same card numbers (replacement cards) are detected. At the R level processing, the cards are read in and, wherever possible, basic range checking is performed to be sure that the input variables fall within physical limits. At this input level the code is able to provide only primitive input checking, since information is available only about the current card. During the initialization, or I level processing, more global range checking is performed; and the code is able to verify self-consistency between cards.

As a minimum, all input will be subjected to four comparative checks: (1) physical/code limits, such as a fuel pellet radius greater than the inner cladding radius, (2) consistency of input, such as a radial node omitted at a material interface, (3) number of words on a card, and (4) variable type. A fifth check, for range of normal use, will also occur during input processing wherever applicable. Input violations of physical and/or code limits, consistency of input, number of words on a card, and variable type will result in an input error but will not abort input processing. Wherever possible, input data that has previously been shown to cause a code abort are now tested, and diagnostic messages issued. Rejected input will be identified and reset to a benign value to allow complete input processing. The selected ranges of allowable input are listed with the card input descriptions in Appendix A. This appendix also contains additional information on the type of checks that will be performed.

Sequential expansion, as found in RELAP5, is used wherever possible. In sequential expansion, sets of data used to specify parameters are followed by an integer, which specifies the range over which the

parameters should be applied. As an example, if a data set can be applied to each axial node of a component, then the integer would be the final axial node over which the data were to be applied. Utilization of sequential expansion significantly decreases the size of an input deck and is a technique which RELAP5 users have applied for many years. Additional examples are available in Reference 2.

3.2 Input Preparation

Attention to detail in preparing, documenting, and checking the input limits errors and provides a valuable model reference for tracking error corrections and subsequent model improvements. By using standardized input format and conventions, input errors are easier to detect. The following sections discuss standard procedures for model documentation and quality assurance, input deck arrangement, and conventions.

3.2.1 Input Deck Arrangement

The code accepts data based on the “card number” specified in the first field on each line of input. For a given card number, the code accepts the input parameters specified in the code manual as sequences of floating point, integer, and alphanumeric entries. On any given card, the data entries must appear in the proper sequence and be separated by one or more blanks. The cards may appear in any order, as long as all required cards and data entries are present. If a card number is duplicated in the input listing, the code identifies it as a “replacement card” and uses the information on the last card entered with that number.

As stated above, the input deck cards may appear in any order. In practice, however, arranging the cards in a logical manner is preferred. At the INEL input decks typically start with the title, job control, and time step control cards. These are followed in sequence by the minor edit requests, trip specifications, hydrodynamic components, heat structures user-input data tables, control variables, and reactor kinetic specifications. An input deck is generally arranged by increasing card numbers when this arrangement is used. Within each of the above groups, data are similarly arranged in order of the card numbers (e.g., the trips are listed in numerical order).

A well-organized input deck includes comment cards that aid interpreting the input from a printed listing. Comments may be inserted through the use of the asterisk (*). On any line, all entries following an asterisk are assumed to be comments. With this format, an analyst will spend a minimum amount of time counting fields and searching through the manual to understand the input.

3.2.2 Model Input Debugging

The input processing routines provide excellent error-checking and error-interpretation capabilities. Input processing error checking is invoked when executing both new- and restart-type problems. All model input errors result in the generation of an informative error message. The presence of one or more input errors results in job termination and a message that the termination was due to input error. As a word of caution, the SCDAP/RELAP5 error-checking functions are primarily intended to check for compliance with the input data requirements. Secondarily, checking is performed for model consistency (e.g., that fuel rod diameter does not exceed pitch). However, the input error-checking function may not uncover basic input errors such as incorrectly specifying a radius of 0.050 m as 0.50 m. Therefore, successful completion of SCDAP/RELAP5 input processing should not be considered a replacement for a quality assurance activity

such as a ‘workbook’, as described in the RELAP5 user guidelines[2].

An efficient method for debugging a new SCDAP/RELAP5 input deck is described as follows. The complete model is first assembled into a single file and the model is executed in either the transient or steady-state modes as specified on card 100. Either the INP-CHK or the RUN option may be selected on card 101. A typical new input deck will likely contain many input errors so the execution will result in generation of a series of error messages. It is common for one actual error to propagate into the generation of multiple error messages. Therefore, the list of error messages generated will in general be much longer than the actual number of errors in the model. The user should read and consider each of the error messages in the order they were generated. This process results in one of the following determinations for each of the error messages: (a) the message clearly indicates an error in the deck and the resolution is clear, (b) the message is found to be caused by the existence of a previous error and is expected to be resolved when the primary error is corrected, and (c) the reason the message was generated is not clear. In practice, the error messages are very informative and the actual input errors are obvious to the analyst. A significant effort can be expended tracing the source of each error message. Instead, it is more efficient to survey the error messages, correct the obvious errors, and again execute the model. As a rule of thumb, only about one third of the error messages generated are caused by actual errors; the remainder are second-generation messages resulting from the primary errors. This iterative process proceeds rapidly to the removal of all input errors. Experience shows that a large input deck that has been entered with moderate care can be debugged with this process in about five iterations.

The iterative debugging process described in the previous paragraph can be much easier if the output of the debugging runs are reviewed on a terminal by an editor capable of searching for data strings. All input error messages are preceded by a string of eight asterisks (********) and the removal of all errors results in the generation of the message “Input processing completed successfully”. The user should be cautioned that even when there are no input error messages (marked by eight asterisks), there may still be input warning messages (marked by eight dollar signs). Although not fatal, these messages may assist in identifying additional errors.

The user should be aware that the input processing is subdivided into several sections of data checking that are performed in sequence. Depending on the nature of the errors found, the job may be terminated at the end of one of the sections before all of the error-checking sections have been executed. In this instance, only error messages for the sections that have been checked will appear. When these errors have been corrected and the checking proceeds to the next section, the number of error messages may increase. In other words, the analyst should realize that in this iterative process the number of error messages may not monotonically decrease.

3.3 Problem Execution

When the input deck has successfully passed input processing, an initial time edit will be generated by the code. If the RUN option is selected, problem execution proceeds from the conditions specified in the initial edit. The initial edit will be identified as zero time for NEW problems and as the time of the restart

edit for RESTART problems.

3.3.1 Time Step and Edit Selections

The problem execution is controlled by the options specified on the 201-299 time step control cards. These cards specify the time step sizes and output features desired as the problem progresses from one time interval to the next. Card 201 specifies these options and the end time for the first time interval, Card 202 for the second time interval, and so on. Subdividing the problem into time intervals facilitates modifying the execution to suit the expected nature of the problem. For example, consider the case of a modeling action (such as closing a valve or tripping a pump) that is of particular interest and may slow the calculation at a given time (say 10 seconds). For this case, a first execution interval might be selected to end at 9 seconds. The second interval might include a reduced time step, and perhaps increased edit and plot frequencies, from 9 to 15 seconds. After 15 seconds, a third interval would then be used to return the time step and edit options to their original values. Note that execution is terminated if the problem time reaches the end of the last interval specified on the 201-299 cards.

For each time interval, minimum and maximum time steps are specified. The code will attempt to execute the problem at the maximum time step. The first time step taken will be at the maximum value. The user is cautioned to use a small maximum time step size when first executing a model for which gross approximations of initial conditions have been specified. Time step size is automatically reduced based on a number of tests. The material Courant limit may not be violated. Mass, fluid property, quality, and extrapolation errors are monitored in each calculational cell and the time step is reduced if errors exceed internally preset limits. The major edit output indicates the criteria and model region causing time step reduction. This indication can be useful for improving model performance.

The code accomplishes time step reductions by repeated division by two until the errors are within acceptable limits, the minimum time step size is reached, or a failure is encountered. The severe accident subcode, SCDAP, now has the ability to impact the time step selection as well. Phenomena which have more impact during severe accident analysis, such as radiation heat transfer, can apply significant stress to the code. These phenomena can now be used to force SCDAP/RELAP5 to repeat the time advancement with a reduced time increment.

3.4 Plot Variables

One of the primary resources for an analyst using the SCDAP/RELAP5 code is the plot file. Severe accident transients, by the very nature, have parameters which are changing rather dramatically with time, and system ‘snapshots’, such as are provided by the major edits, reveal only part of the story. The severe accident analyst is encouraged to make extensive use of the ability to plot parameters from the restart/plot file. Due to the large number of parameters which could appear on the plot file, the philosophy of SCDAP/RELAP5 is to allow the code user to specify the parameters of interest at input. Specification of the parameters of interest are done by use of ‘208’ cards, and the code user is referred to Appendix A of this report

for details on the use of these cards.

3.5 Guidelines for Late Phase Damage Progression

The uncertainties involved in modeling the late phase damage progression make it useful to perform bounding studies on the calculated times of molten pool slumping and failure of the lower head. The areas of modeling with large uncertainty include: 1. strength and configuration of solidified material that supports a pool of molten core material; 2. fragmentation temperature of embrittled fuel rods that are quenched; 3. flow area of break in piping system, 4. configuration of slumping molten material, and 5. heat transfer coefficient between debris and the lower head of the reactor vessel. A parameter for each of these areas of modeling can be defined by the code user so that a series of analyses can be performed to bound the possible behavior of the reactor. This section provides guidelines for the range of values of these parameters in order to calculate the range of possible reactor behavior.

An integer parameter is provided on SCDAP input card 40001100 to provide an estimate of the range of time in which a molten pool slumps to the lower head. If this parameter is set to a value of one, then the molten pool is considered to slump to the lower head whenever material at the periphery of the core has become molten. In this modeling option, solidified material at the periphery of the core is considered to have no strength for supporting a molten pool. This value of the input parameter provides an estimate of the earliest possible time of molten pool slumping. If the parameter on Card 40001100 is set to a value of zero, then the crust supporting the molten pool is considered to always have the strength necessary for supporting a molten pool. The molten pool does not slump to the lower head until its supporting crust at some point is calculated to melt. This value of the input parameter provides an estimate of the latest possible time for slumping of the molten pool. For both values of the input parameter, if the molten pool is calculated to slump, all of the molten material is calculated to slump. The assumption is applied that the initial point of failure of the crust is eroded to a depth sufficient to allow drainage of the entire molten pool. This assumption and the two types of slumping behavior defined by Card 40001100 are an interim solution until a model is implemented for calculating the structural integrity of the crust

The fragmentation temperature of embrittled fuel rods is defined on Card 107 of the RELAP5 input. Embrittled fuel rods are considered to fragment when their temperature decreases to a value less than the fragmentation temperature. The fragmentation temperature is user defined because the code does not have a mechanistic model for fragmentation. The most likely time for embrittled fuel rods to fragment is during the period of rapid temperature change that occurs when the mode of heat transfer at the cladding surface changes from film boiling to nucleate boiling. The thermal stresses in the cladding are maximum during this period of time. An upper bound value on fragmentation temperature is estimated to be the temperature at which the mode of heat transfer at the surface of fuel rods being quenched changes from film boiling to transition boiling. The lower bound of the fragmentation temperature is estimated to be the temperature at which the nucleate boiling mode of heat transfer occurs, which is near the saturation temperature of water. Other mechanisms for fuel rod fragmentation may be possible. If the user identifies one of these other mechanisms being in operation, then a fragmentation temperature appropriate for this mechanism should be defined. The user defined value for fragmentation temperature has no influence on calculated results for the case of severe accidents in which no embrittled fuel rods are cooled below the upper bound value of the fragmentation temperature or in which all of the fuel rods are cooled to temperatures less than the lower bound value of fragmentation temperature If fuel rods with cladding that is calculated to be embrittled are

cooled to a temperature less than the fragmentation temperature, then the fuel rods are considered to disintegrate into porous debris. The upper bound on the calculated extent of core fragmentation is obtained by defining the fragmentation temperature to have its upper bound value. The lower bound on the calculated extent of core fragmentation is obtained by defining the fragmentation temperature to have its lower bound value.

The flow area of a break is defined by the RELAP5 input card for the valve component that represents the break. The break size is estimated to range from 25% to 200% of the flow area of the pipe that broke. Creep rupture calculations must first be requested for each possible location in the reactor piping system at which creep rupture may occur. The locations for which creep rupture is to be performed are defined on RELAP5 Cards 21000110 and 21000000. Then after the calculations have identified the time and location of the first creep rupture, the calculations are repeated with a break being defined for the location with a creep rupture.

Molten material that slumps from a molten pool may be represented as either an intact stream as it slumps or breaking up into small droplets. The representation of the slumping molten material is defined on RELAP5 Card 103N. If the stream of molten material is represented as an intact stream, then heat is not transferred from the molten material as it slumps. The molten material is at the same temperature when it impacts the lower head as it was when it was in the molten pool in the core region. In addition, the material that slumps to the lower head is defined to have no porosity. As a result, the lower head of the reactor vessel may heatup rapidly. If the molten material breaks into small droplets as it passes through liquid water, then the molten material is cool when it impacts the lower head. In addition, the material is considered to have open porosity that can be filled with water. The heat transferred from the small droplets of molten material to water may cause a significant increase in pressure in the primary coolant system. If the user defines breakup of the slumping material but the lower plenum of the vessel is not filled with water, then the slumping stream of material is considered to remain intact and not breakup.

The gap heat transfer coefficient between debris in the lower plenum and the lower head that supports this debris is defined on the COUPLE Card 5ccc1101. The gap heat transfer coefficient may range from 500 W/m^2K to 10,000 W/m^2K (Reference 14). The upper bound value of the gap heat transfer coefficient results is a more rapid heatup of the lower head than the lower bound value.

3.6 Code Diagnostics

In the past, SCDAP/RELAP5 has been considered a research tool by both development staff and user community. As such, it was considered more significant to implement state of the art core damage progression models, than to develop the user interface. However, the developing maturity of the code now requires that the code-user interface be easier to understand. A significant effort has been expended to eliminate (a) code failures without error messages, (b) confusing diagnostics sent only to the screen, and (c) confusing error or warning messages. A survey of international code users was used to identify confusing diagnostics and code errors that lead to abnormal aborts. While this will undoubtedly be an on-going effort, with additional areas of improvement identified by the user community, this effort has increased the quality of diagnostic information.

In order to eliminate code stops without error messages, stop statements or calls to abort routines were removed and, where necessary, replaced with statements setting a logical variable, ‘fail’, to true. This action allows the completion of the current time step, and then forces a major edit and a graceful exit. Where possible, logic to repeat a time advancement with a smaller time step has been implemented. Also, error diagnostics were added to identify where the failure occurred. Representative changes were successfully tested to ensure that time-step completion would occur and that the expected messages would result. Many users complained about the nonconvergence messages from the equation solver DSGEDR, which is used to solve the molten material relocation equations. This problem has been resolved with the implementation of the automatic time advancement repetition capability, which now merely repeats the time advancement with a reduced time step.

Major edit output has been improved. The information needed to define the molten pool and debris regions has been reduced in length, and the number of tables showing core configuration has been reduced from six to one. A new output format replaces references to variable names with clear descriptions for output parameters.

SCDAP/RELAP5 generates one- or two-line messages whenever significant changes in core conditions occur. These messages, denoted real time messages, are printed as soon as the condition is changed, and therefore will appear between major edits. Since the information in these messages are sometimes difficult to interpret, an effort is being made to clarify and/or eliminate them. However, changing this output is difficult, because these write statements number in the hundreds and are spread throughout the code. Changes have been implemented to add subroutine identifier statements to all lines of this output for identification. Statements that are printed following a single major event are grouped under the header ‘core degradation event’. Also, times for these events are now printed with greater number of significant digits, to ensure that timing is known; and the times are listed as ‘event time’ to help the user when searching output with a text editor. In the future a table will also be generated during the major edit, which will summarize core degradation event timing since the last edit.

The user community is encouraged to identify confusing, unneeded, or missing output and to pass their suggestions to the development staff. An on-going effort will be made to improve the user interface to SCDAP/RELAP5.

3.7 Steam Starvation Limits

During the developmental assessment calculations documented in Volume V, an anomaly was observed in some experimental analyses. Although the calculated cladding surface temperatures closely matched the measured temperatures, in some experiments the bundle hydrogen production was over-predicted. Since, obviously, the physics of oxidation are well understood, either a systematic error was being introduced into the experimental data, or a phenomena not currently being modeled is affecting the oxidation process in some experiments.

One of the most basic assumptions used in the hydrodynamic portion of SCDAP/RELAP5 is that the fluid/gas mixture within a hydrodynamic volume is homogeneously mixed within each hydrodynamic volume. While this assumption is valid in most cases, it is conceivable that during rapid oxidation a large quantity of noncondensable gas could be generated sufficiently quickly that it will inhibit the transport of steam to the oxidizing surface. The model for this limitation is described in detail in Volume II, but

essentially applies a factor of the ratio of the partial pressure of steam to the local bulk pressure as a multiplier on the steam availability. The limit then only becomes effective as the noncondensable quality becomes high. Any component may then be oxidation limited either by the availability of steam or the diffusion of steam through a noncondensable.

It has become a standard practice for analysis of experimental facilities, and therefore is recommended as a user guideline, to take a group of fuel rods that would normally be represented as a single SCDAP/RELAP5 component, and to characterize that group with two components, identical except for oxidation limit. As an example if the user has a group of n fuel rods which respond identically, and can therefore be represented with a single component, then it is recommended that the user characterize their response with two components, one component representing n-1 fuel rods with the diffusion limit active, and one component representing a single fuel rod with the steam starvation limit active. The diffusion limited component can then simulate the behavior of the bulk of the core, and can be used to predict core average parameters, hydrogen production for example. The single fuel rod, with oxidation limited only by steam starvation, can then represent the behavior of the instrumented rods, and can be used to predict maximum cladding temperature. This procedure should bound the effects of noncondensables.

It should be noted that the model to limit oxidation by the diffusion of steam through noncondensables is the default, with steam starvation limit obtained only on request through the 4ccc4001 - 999 cards, as described in Appendix A.

3.8 Nested Radiation Enclosures

The ability of SCDAP/RELAP5 to model radiation transfer across nested radiation enclosures has been considerably enhanced. In the past each core component could exist in one and only one radiation enclosure. Additionally two sided components, such as the shroud component, could only radiate heat from the inner surface. However, analyses such as that of the CORA experiments, where an insulated flow shroud was surrounded by a high-temperature shield, required the ability to model nested radiation enclosures. In these experiments, heat from electrical heater rods is transferred to the inner surface of the insulated flow shroud, then conducts through the shroud to its outer surface, and then is transferred by convection and radiation to the inner surface of the high-temperature shield. The model of radiation heat transfer requires the representation of two radiation enclosures; one representing the space inside the insulated flow shroud and another representing the space between the outer surface of the flow shroud and the inner surface of the high-temperature shield. Both of these enclosures are connected with the insulated flow shroud.

3.9 Cold Wall Effects on Core Damage Progression

Thermal and hydraulic analyses of nuclear reactor core response under accident conditions have been successfully carried out in the past with models that assume that large fractions of the core behave in a similar manner. Such analyses typically use a hot channel approach, which assumes that a significant group of fuel rods (typically on the order of one-third of the core) represent a group of the highest power fuel rods, while the remainder of the core is modeled as a second group of average fuel rods. Each group is then sectioned into it’s own hydraulic channel. Such a modeling approach forces two assumptions on the analysis. The first assumption is, obviously, that the fuel rods in a large fraction of the core behave in an identical manner. The second assumption, both more subtle and more significant to this discussion, is that

each of the fuel rods within the group acts independently of every other fuel rod within that group. This 'hot channel' analytical technique has been successful in the past because this second assumption has been reasonably valid for thermal-hydraulic phenomena. Such is not the case, however, for the phenomena associated with early phase severe accident conditions.

Phenomena associated with the core response to early phase severe accident conditions, particularly relating to the mechanical response of the fuel rods, are strongly dependent on conditions experienced by an adjoining fuel rod. For instance, if a fuel rod experiences an azimuthal temperature variation around it's cladding, due to a neighboring cold wall, while an adjacent fuel rod encounters cladding ballooning conditions, then coolant flow in the unit cell with the ballooned rod is diverted into the unit cell without ballooning, causing additional cooling of the adjacent fuel rod, thereby making it less likely to experience ballooning conditions. Similar conditions are generated by any phenomena which causes flow diversion, such as during melt relocation. The inability of the typical hot-channel analytical technique to track this type of dependent behavior forces the analysis to predict co-planar blockages and over-estimates the reduction in flow area. The flaw in this analytical technique has become particularly obvious for experimental facilities which examine both early and late phase severe accident behavior. These facilities, by their very nature, are forced to expose the experimental apparatus to relatively large radial temperature gradients, with cold wall effects generated by facility boundaries, thermal flow shrouds, and by instrumentation.

The SCDAP/RELAP5 code, developed for best-estimate transient simulation of light water reactor coolant systems during a severe accident, has been extended to allow the code user to specify multiple parallel hydraulic flow channels within a single radiation enclosure. This extension will allow the code user to avoid the 'hot channel' analysis technique, and will permit the use of parametric studies to evaluate the cold wall effect on core damage progression. This extension will allow the user to perform sensitivity studies of cold wall effects, and parametric nodalization studies. The effects of cold walls on core damage progression has been shown to be potentially significant, particularly in experimental facilities, where significant radial temperature gradients have been observed.

Additionally, it appears that under some conditions, plant analysis models that have been successfully used to examine the thermal hydraulic behavior of a system under accident conditions can be inappropriate for severe accident analysis, particularly when system aspects, such as the presence of one or more cold walls, are not appropriately modeled. The typical hot-channel thermal-hydraulic analysis that has been used in the past, inherently assumes that each fuel rod in each channel behaves independently. This assumption can impose phenomena such as co-planar blockage and over-prediction of flow area reduction when phenomena such as cold-wall effects, cladding deformation, or melt relocation are being experienced.

3.9.1 Noncondensable Model

The noncondensable model is implemented by specifying a noncondensable gas type on control card 110 and indicating a noncondensable quality on one or more volume initial condition cards. A mixture of noncondensable gases may be specified by indicating more than one gas type on card 110 and specifying their mass fractions on card 115. It should be noted that only one noncondensable gas mixture may be used in a problem, although the fractions of each gas type may change in each hydrodynamic volume, and the noncondensable gas must be hydrogen (or include hydrogen in the case of a mixture). This means that if

nitrogen is present in one part of the system and hydrogen is present in another, then the system has a mixture of hydrogen and nitrogen, with the mixture consisting of 100% nitrogen and 0% hydrogen in one location and a mixture of 0% nitrogen and 100% hydrogen in another.

The noncondensable model assumes the gas is tracked with the vapor phase. Furthermore, the resulting gas-steam mixture is assumed to be isothermal (i.e., the gas and steam are in thermal equilibrium). A total pressure is calculated for the gas-steam mixture; the partial pressure of steam is available as a standard output variable.

3.10 BWR CHANNEL BOX USER GUIDE

This Chapter describes the input data that the user must specify on SCDAP input cards for the BWR control blade and channel box component. Also, information is provided to help the user interpret the printed output.

3.10.1 BWR Blade/Box Cards

The specific SCDAP input cards for the BWR control blade and channel box component are documented in Appendix A. This Section provides additional information to help the user prepare data for the input deck. All descriptions in this Section refer to the new SCDAP input format with RELAPS-style card numbers (see Section A. l of Appendix A).

Cards 40003000, 40003100, and 40003200 contain parameters that are common to all BWR blade/box components. These cards should appear not more than once in an input deck. Default values (see Appendix A) are provided for each of the parameters on Cards 40003000, 40003100, and 40003200. If no other information is available, the user should invoke these default values by omitting Cards 40003000, 40003100, and 40003200.

The failure (liquefaction) temperatures on Card 40003000 are used to account for the effects of eutectic interactions between B_4C/stainless steel and stainless steel/Zircaloy. Eutectic interactions are modeled by using failure (liquefaction) temperatures that are less than the melting temperatures of the pure materials.

The metal/water reaction parameters on Card 40003100 affect the B_4C and Zircaloy oxidation calculations. The user must specify a maximum fraction of B_4C in each node that can react. This maximum fraction is used by the advanced $B_4C/H_2/H_20$ chemistry package to control the mass of B_4C available for the chemical equilibrium calculations. The user can also select different Zircaloy oxidation correlations for the low and medium temperature regions. These Zircaloy oxidation correlations are described in Section 2.5 of this report.

On Card 40003200 (relocation parameters), the user must specify two mass fractions of oxides (stainless steel and ZrO2) that are carried along with the underlying pure materials when they melt. The user must also specify two heat transfer coefficients between molten material (stainless steel and Zircaloy) flowing downward and the underlying solid structures. (The model does not currently include correlations for these heat transfer coefficients.) The recommended default values (see Appendix A) for these heat

transfer coefficients were determined empirically from ORNL analyses of the DF-4 and CORA experiments.

The control blade and channel box dimensions specified by the user on Cards 4ccc0200 and 4ccc0300 are sketched in Figure 3-1. The actual control blade radial dimensions shown in the top of Figure 3-1 are converted by the model into the equivalent slab geometry shown in the bottom of the figure. The equivalent slab thicknesses are calculated so that the cross-sectional area of each layer in the equivalent slab geometry is identical to the cross-sectional area in the actual geometry. The distance between the channel box and the first row of fuel rods (dimension 7 in Figure 3-1)is used in the relocation calculations to determine when the region on the fuel-bundle side of the channel box is blocked..

As is the case for all other SCDAP components, the internal modeling for the BWR blade/box component is performed using a local set of dimensions that describes the single blade/box structure shown in Figure 3-1. However, the BWR blade/box component can be used to represent many copies of this individual blade/box structure by specifying the value on Card 4cccO100. If a value of 1 is specified on Card 4cccO100, then the component will perform calculations for half of a control blade and two channel box segments with lengths as indicated in Figure 3-1 (dimensions 8 and 9).

The geometric view factors specified on Card 4ccc0300 are for radiation between the channel box and the control blade, which is modeled internally by the BWR blade/box component. These geometric view factors must be calculated by the user using the geometry sketched at the bottom of Figure 3-1. The sense of direction is from the channel box to the control blade, i.e., the view factors are based on the areas of the channel box segments.

Initial conditions for the BWR blade/box component are specified on Cards 4cccO500 and 4ccc0601-99. The three oxide thicknesses on Card 4ccc0500 apply to all axial nodes. The initial stainless steel oxide layer must be specified nonzero because this value is used as a denominator in the stainless steel oxidation calculations. This restriction does not apply to the initial ZrO2 layers; they may be specified zero. The initial control blade temperatures specified for each axial node on Cards 4ccc0601-99 (Word 1) apply to all three radial nodes. The initial channel box temperatures specified for each axial node on Cards 4ccc0601-99 (Word 2) apply to both channel box segments.

If fuel rod or electrically-heated simulator rod components can receive molten material from a B~YR blade/box component, then this information is specified on Cards 4ccc0701-99 and 4ccc0801-99. Cards 4ccc0701-99 apply to radial spreading from channel box segment No. 1 while Cards 4ccc0801-99 apply to segment No. 2. The mass fractions of molten material from channel box segment Nos. 1 and 2 are used to determine how molten material is proportioned between multiple fuel or simulator rod components that are located adjacent to the same channel box segment.

The mass fractions on Cards 4ccc0701-99 and 4ccc0801-99 can be used to represent radial spreading that occurs initially into the first row of fuel rods and later into the remainder of the fuel bundle. For example, assume the BWR fuel assembly shown in Figure 3-1 is modeled with SCDAP using one BWR blade/box component (No. 1) and three fuel rod components (Nos. 2, 3, and 4). If the following cards are specified for BWR blade/box radial spreading:

*crd.no comp.no frac.segl

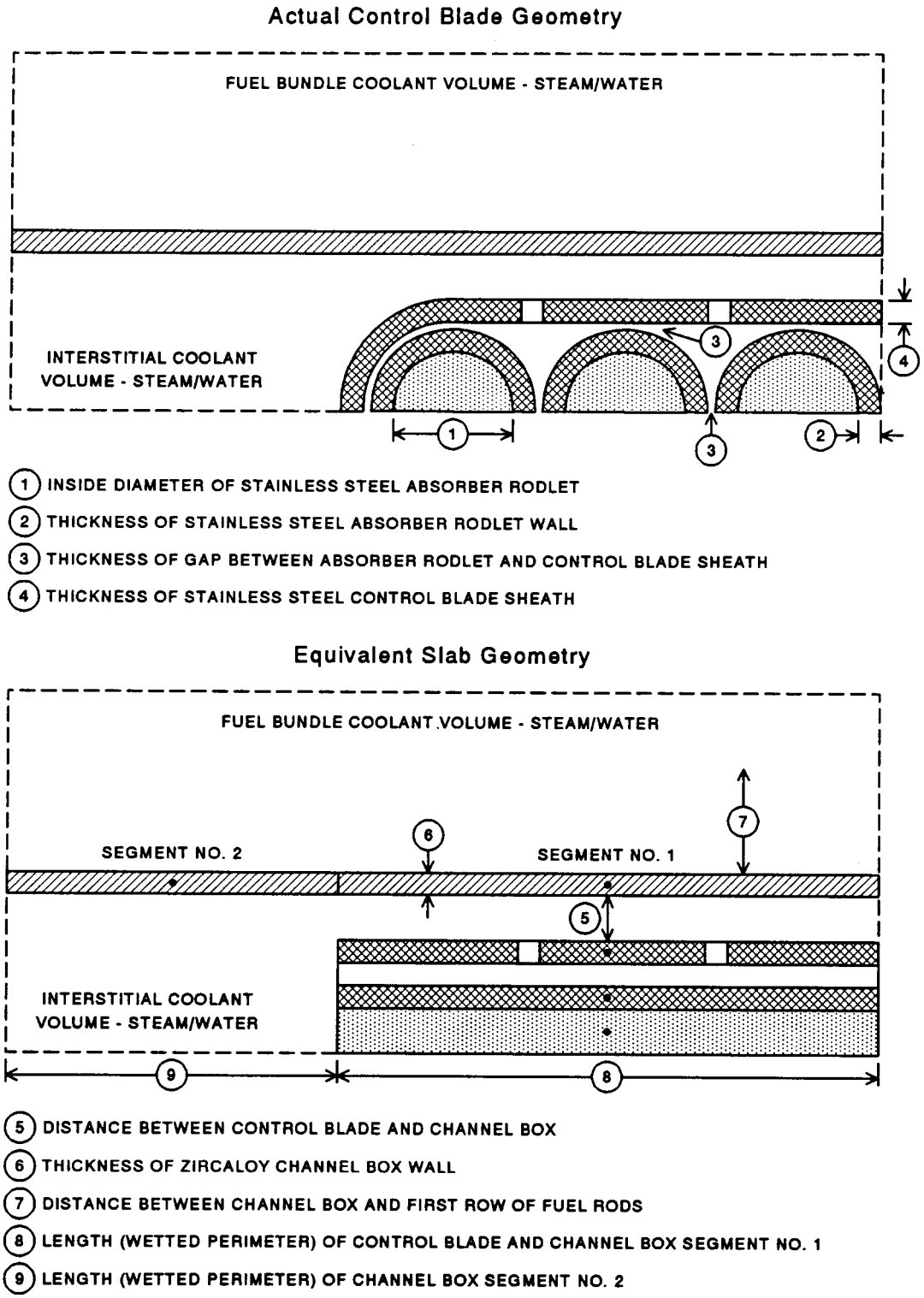

Figure 3-1. BWR control blade and channel box dimensions specified by the user.

```
40010701 2     0.999
40010702 4     0.001
*crd.no comp.no frac.seg2
40010801 3     0.999
40010802 4     0.001
```

then almost all molten material from channel box segment No. 1 (99.9%) will initially be received by fuel rod component No. 2 and almost all molten material from channel box segment No. 2 (99.9%) will initially be received by fuel rod component No. 3. However, the BWR blade/box relocation logic adjusts the mass fractions on Cards 4ccc0701-99 and 4ccc0801-99 when one of the fuel rods becomes blocked at an axial level by cohesive, rubble, or molten debris. Referring to the above example, after fuel rod component No. 2 becomes blocked at an axial level by debris, the mass fraction on Card 40010701 (0.999) is changed to 0.0 and the mass fraction on Card 40010702 (0.001) is increased to 1.0. Subsequently, all molten material from channel box segment No. 1 will be received by fuel rod component No. 4..

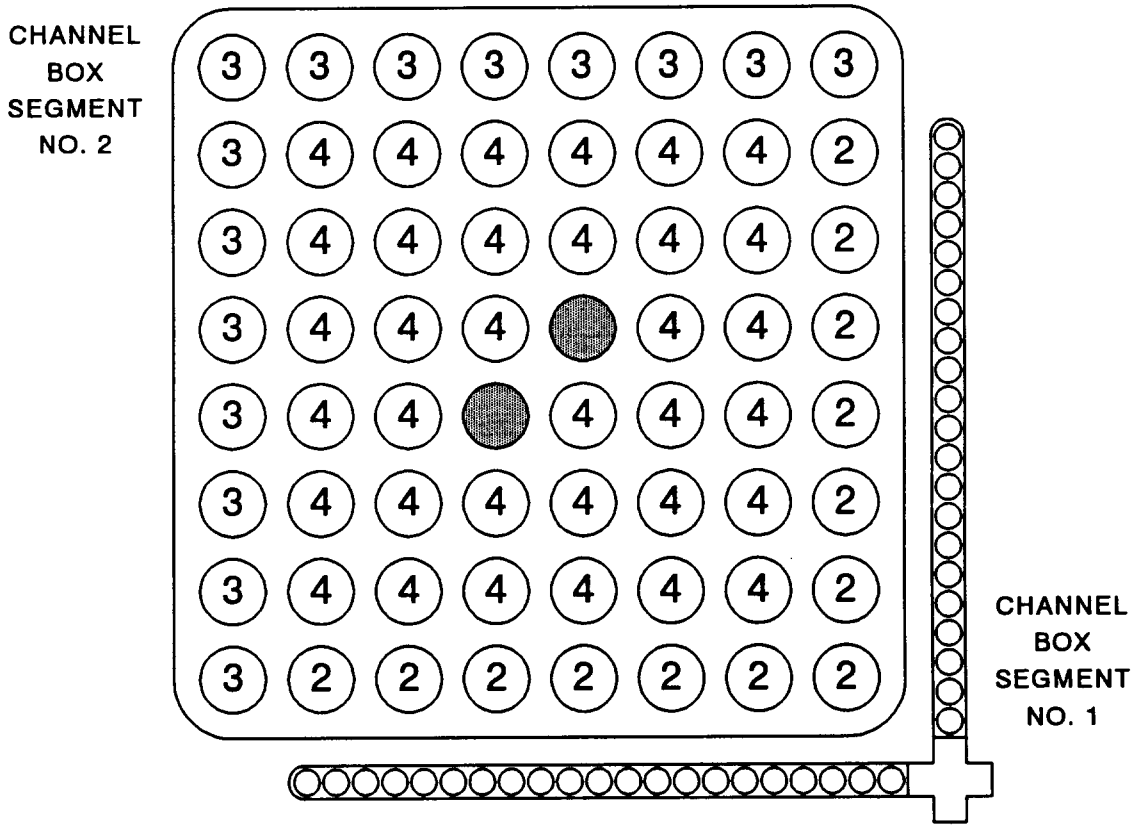

Figure 3-2. Example arrangement of fuel rod components.

3.10.2 Radiation Enclosure Cards

Each BWR blade/box component must be associated with two SCDAP radiation enclosures. One enclosure is for the fuel-bundle side of the channel box and the other is a “dummy” enclosure for the interstitial side of the channel box. Radiation calculations on the fuel-bundle side of the channel box are

performed within the SCDAP radiation model using independent surfaces to represent the two channel box segments. The dummy enclosure on the interstitial side of the channel box is not actually used to perform radiation calculations between the channel box and the control blade (these calculations are performed internally by the BWR blade/box model), but this enclosure is needed to initialize properly the hydrodynamic calculations for the interstitial volume. If the user does not define both SCDAP radiation enclosures for each BWR blade/box component, an error message is printed and execution is terminated after the completion of input processing.

When a BWR blade/box component is included within a radiation enclosure, the view factors and path lengths for that enclosure must be specified by the user on Cards 4ccc1001-99 and 4ccc1101-99. Because the two segments on the fuel-bundle side of the channel box are treated independently, view factors between channel box segment Nos. 1 and 2 can be calculated and specified, if necessary.

In the radiation enclosure section of an input deck (see the following example), the component number of a BWR blade/box component must be listed three times on Cards 4ccc1000. The first two BWR blade/box entries must be consecutive and are part of a radiation enclosure that represents the fuel-bundle region. These first two entries refer to segment Nos. 1 and 2, respectively, on the fuel-bundle side of the channel box. The third BWR blade/box entry must be on a separate Card 4ccc1000 that represents the dummy enclosure for the interstitial side of the channel box. The input cards that define this dummy enclosure must follow the cards that define the fuel-bundle radiation enclosure.

For example, suppose there are two components in a SCDAP input deck and component No. 1 is a fuel rod and component No. 2 is a BWR blade/box. To model radiation between the fuel rods and the two channel box segments, the user must define enclosure No. 3 for the fuel bundle side of the channel box, followed by dummy enclosure No. 4 for the interstitial side of the channel box, using the following format:

```
*crd.no
40030000
*card. no
40031000
*crd.no
40031001
40031002
40031003
*crd.no
40031101
40031102
40031103
*crd.no
40040000
*crd.no
40041000
*crd.no
40041001
*crd.no
40041101

name type fuel      bundle comp.nos 1 2 2 view . factor 0.2818 0.9400 0.2094 path . length 0.001 0.005
```

```
        0.015 name dummy comp.nos

2

view.factor 1.0 path.length 0.0

0.6782
0.0
0.2267

0.005 0.0 0.015 type bundle 0.0400
0.0600
0.5639

0.015
0.015
0.010
```

Component No. 2 (BWR blade/box) is listed twice on Card 40031000 and once on Card 40041000 of the above example. The view factor and path length arrays for enclosure No. 3 have two sets of entries for component No. 2. The second value on Card 40031001 (0.6782) is the view factor from the fuel rods to channel box segment No. 1. The third value on Card 40031001 (0.0400) is the view factor from the fuel rods to channel box segment No. 2. The third value on Card 40031002 (0.0600) is the view factor from channel box segment No. 1 to channel box segment No. 2. For enclosure No. 4, the view factor on Card 40041001 and the path length on Card 40041101 are dummy values.

3.10.3 Minor Edit Requests

The BWR blade/box variables that can be printed at “Minor Edits” or written to the restart-plot file are listed below. Although the variable names are identical to those used for other SCDAP components, the definitions listed below apply only to BWR blade/box components. These variables are “Expanded Edit/ Plot Variables” and, therefore, are not written to the restart-plot file by default. To write these variables to the restart-plot file, a RELAP5 Card 2080xxxx with the appropriate name and index listed below must be added to the input deck. For all variable names listed below, “Index” is defined as: ii = radial node number, kk = axial node number, and jj = SCDAP component number.

Table 3-1. BWR blade / box variables.

Name	Index	BWR blade/box definition
CADCT	*iikkjj*	*Temperature (K) at radial node ii and axial node kk of component jj. For a BWR blade/box component, valid values of radial node ii are l14 (see Figure 3-1 for locations).*

Table 3-1. BWR blade / box variables. (Continued)

Name	Index	BWR blade/box definition
DAMLEV	*kkjj*	*Level of damage (unitless) at axial node kk of component jj. For a BWR blade/box component, this indicates when the channel box wall has failed and a flow path has opened between the interstitial and fuel bundle coolant volumes* *.0.0 = Both channel box segments intact* *0.1 = Channel box segment No. 1 gone* *0.2 = Channel box segment No. 2 gone* *0.3 = Both channel box segments gone*
H20XD2	*kkjj*	*Total hydrogen production rate (kg/s) at axial node kk of component jj. For a BWR blade/box component, this is the total hydrogen from the control blade and both sides of the channel box.*
OXDEO	*kkjj*	*Frozen crust thickness (m) on the interstitial side of channel box segment No. 2 at axial node kk of component jj.*
RCI	*kkjj*	*Equivalent thickness (m) of the intact control blade sheath at axial node kk of component jj.*
RCO	*kkjj*	*Frozen crust thickness (m) on the control blade at axial node kk of component jj.*
ROCRST	*kkjj*	*Thickness (m) of the intact channel box segment No. 1 at axial node kk of component jj.*
RPEL	*kkjj*	*Thickness (m) of the intact channel box segment No. 2 at axial node kk of component jj.*
RULIQ	*kkjj*	*Equivalent thickness (m) of the intact absorber rodlet (B_4C and stainless steel) at axial node kk of component jj.*
WREMUO	*kkjj*	*Frozen crust thickness (m) on the fuel-bundle side of channel box segment No. 2 at axial node kk of component jj.*
WREMZR	*kkjj*	*Frozen crust thickness (m) on the fuel-bundle side of channel box segment No. 1 at axial node kk of component jj.*

The locations of the BWR blade/box radial nodes for temperature variable CADCT are shown in Figure 3-1. The intact structures of the control blade and the channel box are at radial locations 2-4, 6, and 12. The temperatures defined for radial locations l and 14 are average surface temperatures that are used as boundary conditions for the RELAP5 hydrodynamic calculations. The temperatures at the other radial locations (5, 7, 8-11, and 13) have unique values only when those nodes are blocked and filled with relocated material; otherwise they are set equal to the temperature of the adjacent intact structure..

The variable DAMLEV is defined so that it can be used to control gas flow between the interstitial and fuel bundle coolant volumes after the channel box wall has failed. This is accomplished by using RELAP5 servo valve components with valve areas calculated by control system components based upon the values of DAMLEV.

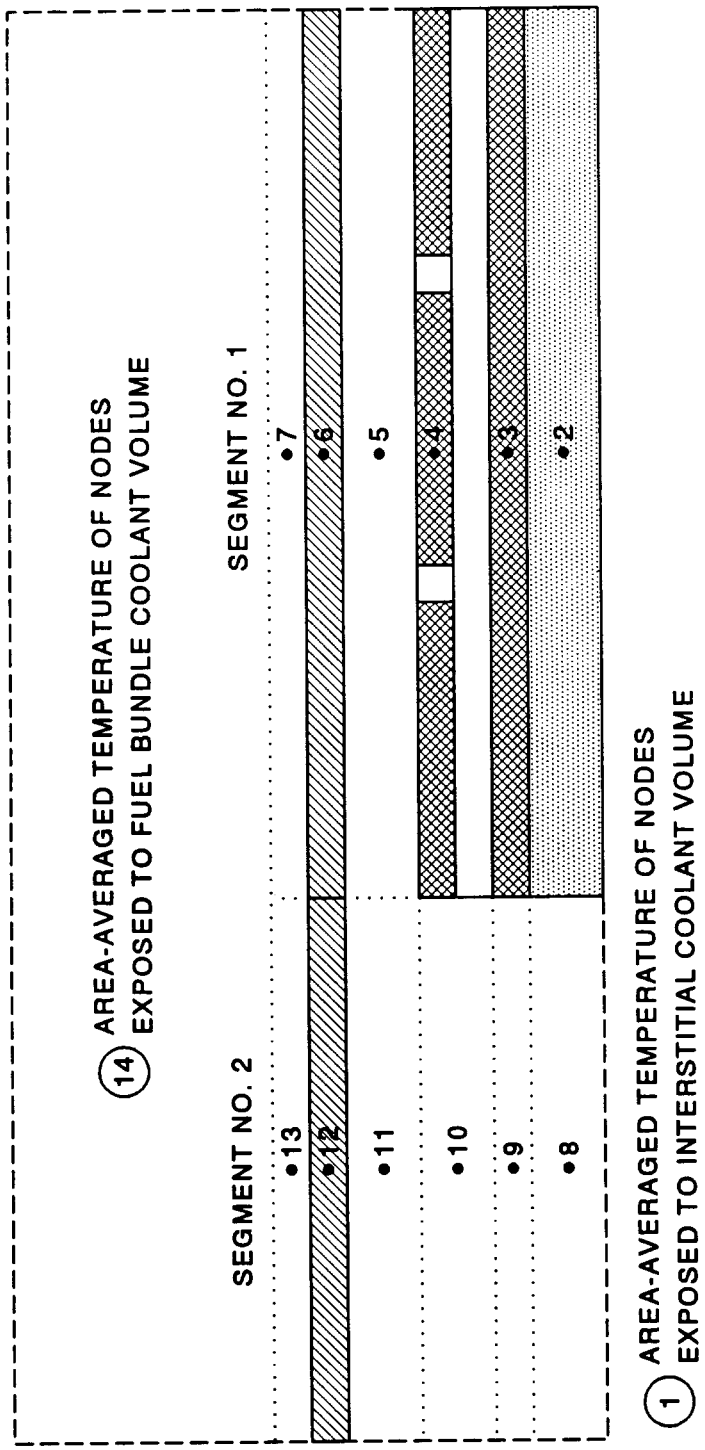

Figure 3-3. Radial node numbers used to print BWR blad/box temperatures at minor edits.

The locations of the intact blade/box and frozen crust thickness variables are shown in Figure 3-1. These eight variables can be used with the Nuclear Plant Analyzer (NPA) program to generate animated drawings depicting the melting and downward relocation of the control blade and channel box structures. This is accomplished by using NPA "Deform Boxes" to represent each variable at each axial node. Note that variables RULIQ and RCI are the equivalent (not the actual) thicknesses of the intact absorber rodlet (B_4C and stainless steel) and the control blade sheath. The user can determine the initial values of RULIQ and RCI using information printed during the processing of input data (see Appendix B) so that the NPA Deform Boxes can be initialized properly..

3.10.4 Restart Calculations

A BWR blade/box calculation can be continued from a previous calculation by specifying the problem type on RELAP5 Card 100 as "restart" and the appropriate restart number on RELAP5 Card 103. Information from the restart-plot file is used to initialize the variables for BWR blade/box components.

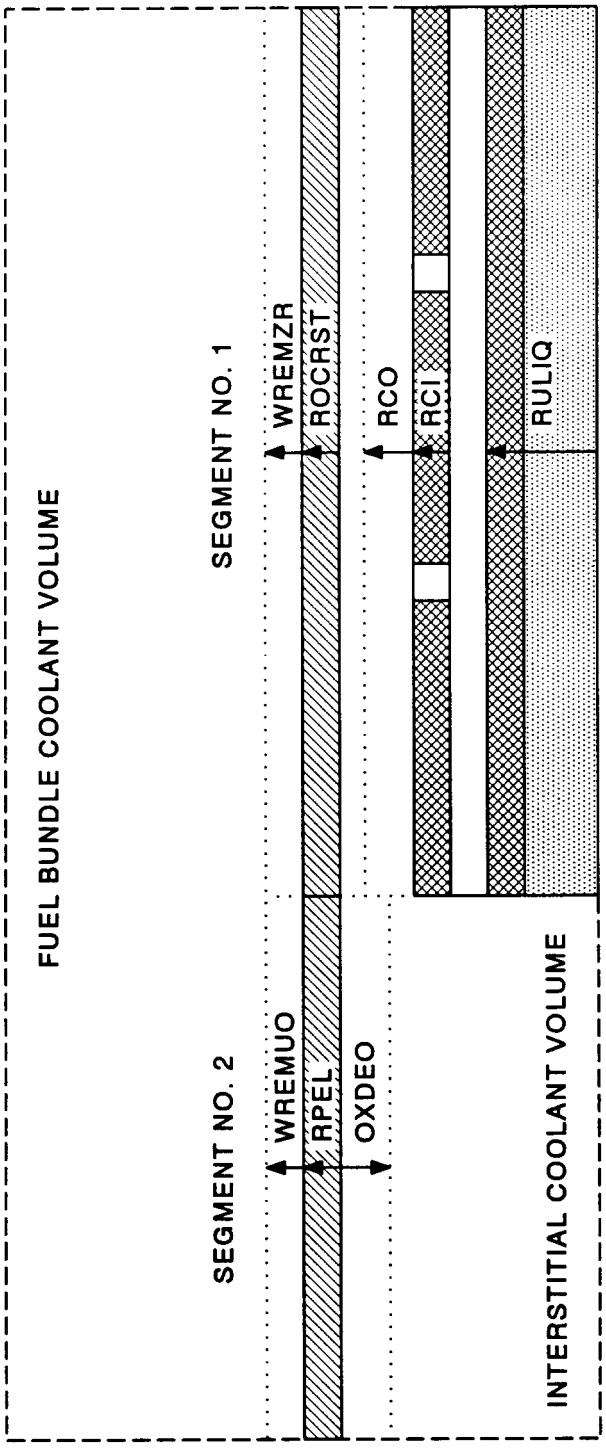

Figure 3-4. Locations of intact blade/box and frozen crust thickness variables.

4. NODALIZATION GUIDELINES

A significant amount of work has been done to characterize the number of volumes/nodes which are appropriate for modelling components in severe accident analyses. The user is referred to Reference 2 for additional details on modelling ex-vessel components, and to Volume V of this report for the details of a core nodalization study which has been performed for both PWR's and BWR's.

4.1 Core Nodalization Guidelines

Requirements for the nodalization of the reactor core for a severe accident analysis is significantly different from that needed for a comparatively simple hydraulic analysis. Section 3.9 discussed the reasons why a one- or two-channel analysis technique, which has been successful for analysis of thermal-hydraulic phenomena, is not appropriate for the phenomena associated with early phase severe accident conditions.

The nodalization of the core with five radial segments and ten to twenty axial nodes is considered to result in an adequate calculation of the behavior of the core during late phase damage progression. The nodalization sensitivity study presented in Volume V of this report showed that the calculated time of slumping of the molten pool may be 7% later using twenty axial nodes instead of ten axial nodes. The nodalization sensitivity study also showed that for the case of surge line rupture being ignored(high pressure case) the use of three radial segments instead of five radial segments results in an incorrect calculation of the location in the core at which late phase damage progression begins and results in a significantly earlier calculated time of beginning of late phase damage. Each radial segment should contain one SCDAP component to represent the fuel rods in that segment and one SCDAP component to represent the control rods/control blades in that segment. For PWRs, each radial segment should contain one RELAP5 control volume that represents the fluid in that segment only. For BWRs, each radial segment should contain two RELAP5 control volumes, one that represents the fluid flowing through the fuel assemblies and one that represents the fluid flowing between the fuel assemblies. Each RELAP5 control volume should be divided into as many subvolumes as axial nodes in the SCDAP components, and the subvolumes should overlay the axial segments of the SCDAP components. The reader is referred to the nodalization study documented in Volume V of this report for additional details.

4.2 Ex-vessel Example Nodalizations

This section provides example SCDAP/RELAP5 nodalizations for PWRs. The purpose of this section is to provide guidance for ex-vessel nodalization that may be used for analyzing a wide variety of small break LOCAs and operational transients. The user is cautioned that no model is generally applicable for simulating all transient scenarios. Care should be taken so that modeling and nodalization are appropriate for the particular application.

For economic reasons, the numbers of hydrodynamic cells and heat structure mesh points in general should be minimized. The computer run time needed to execute a problem simulation is determined almost completely by the number of hydrodynamic cells in the model. The number of heat structures generally increases in tandem with the number of cells. Therefore, a major economic benefit is gained by limiting the number of hydrodynamic cells in a model. Some additional economic benefit may be obtained by minimizing the number of mesh points within the heat structures. Limiting the number of other model features (such

as trips and control variables) provides only minimal economic benefits. An additional motivation for employing the largest calculational cells possible, is that when small cells are used, the time step size is reduced as a result of the material Courant limit. The Courant limit, discussed in Reference 2, limits the calculational time step based on the ratio of cell length to fluid velocity.

The process of minimizing model size must, however, always consider the phenomena to be modeled; minimizing must not proceed past the point where important phenomena are excluded from the simulation. This consideration is complicated, however, because the importance of phenomena varies from one region of the model to another and is strongly affected by the transient to be simulated. For example, the important model regions and simulation phenomena for small and large break loss-of-coolant accidents are dramatically different; therefore, appropriate modeling for these two sequences varies dramatically.

In summary, the modeler should select the minimum number of hydrodynamic cells and heat structure mesh points needed to calculate the important phenomena for the simulated transient. This guidance suggests that a general model (i.e., one that is to be used to simulate many different types of transients) should contain sufficient noding detail for all phenomena anticipated. If the important phenomena are uncertain, a detailed noding scheme should be employed. Conversely, if the important phenomena are well known, nodalization of the non-critical model regions may be simplified. If sufficient time and funds are available, it is recommended that a general model of a reactor system be assembled first. Analysis using the general model will then provide the information needed to determine what model simplifications are appropriate. The following sections provide additional guidance concerning hydrodynamic cell and heat structure sizing. General suggestions for appropriate noding may be inferred from Section 4.

4.2.1 Ex-Vessel Hydrodynamic Cell Size

As discussed above, large hydrodynamic cell sizes should be used for economic reasons. However, in each region of the model, the detail of the calculational cells must be sufficient to allow the simulation of important regional thermal-hydraulic phenomena. As a starting point, cell lengths for ex-vessel hydrodynamic volumes of 1 to 3 m (3 to 10 ft) are recommended in phenomena-dominating regions (e.g., pressurizer, and steam generator) of a light water reactor model. Cells of much longer lengths are appropriate in less important regions of the model (e.g., the feedwater train and steam lines). The cell sizes presented in these applications may be taken as guideline recommendations for modeling light water reactors. For totally new applications or where the calculation results may be particularly sensitive to the model discretization, a convergence study is recommended to ensure that a proposed nodal layout is adequate.

Good modeling practice includes blending the transition from regions of small cells to regions of large cells. For this blending, it is recommended that the volumes of adjacent cells not differ by more than an order of magnitude.

Other considerations affecting cell size selection are the locations of natural boundaries, flow connections, and instruments within the prototype fluid system. Good modeling practice includes placing junctions at natural fluid system boundaries and at flow loss features (such as support plates, grid spacers, bends, and orifices). Using this practice, the flow loss is placed at the proper location with respect to the fluid volumes. For similar reasons, the placement of junctions at flow connection points is a good practice. Cell size selection should also consider placing model features at prototypical instrument locations (e.g., placing a cell

center at the location of a pressure tap or a junction at the location of a flow meter). This practice facilitates the use of the code output because the calculated and measured data are directly comparable without further effort.

4.2.2 Steam Generator Primary Nodalization

The nodalization for the primary side of two types of steam generators are presented, U-tube steam generators (UTSG's) and once-through steam generators (OTSG's).

4.2.2.1 U-Tube Steam Generator Nodalization. Standard INEL nodalization for one of the primary coolant loops is shown in Figure 4-1. Pipe 408 represents the many thousands of steam generator tubes of a U-tube steam generator. Representing the steam generator tube primaries with an 8-cell pipe component is a nodalization scheme that compromises between calculational fidelity and expense. This scheme has proven is generally useful, however the modeler should individually consider the nodalization requirements for the problem to be modeled. The tube nodalization scheme shown may not be sufficiently detailed to model phenomena associated with reflux cooling and greatly reduced secondary-side levels. Branch 410 represents the steam generator outlet plenum. Modeling of the steam generator secondary region is described in the section 4.2.3.

Heat structures are employed to model the hot and cold leg piping walls, the steam generator plena heads, the plena separation plate, the tubesheet, and the steam generator tubes.

4.2.2.2 Once-Through Steam Generators. The OTSG is a counterflow heat exchanger that employs straight tubes. The standard INEL OTSG nodalization is shown in Figure 4-2. Components 116 and 125, represent the OTSG inlet and outlet plena, respectively. Single-sided heat structures represent the significant metal structures (such as the steam generator heads and the tubesheets). Reactor coolant flows downward through the insides of the tubes; 8-cell pipes 120 and 121 represent the tube primaries. Pipe 120 represents 90% of the OTSG tubes, pipe 121 represents the other 10% (the reason for separating the tubes in this manner is discussed below). Two-sided heat structures model the tube walls.

On the secondary side, the downcomer region is modeled with 4-cell pipe 305. Main feedwater enters the downcomer at the upper end of this component. Single-sided heat structures represent the steam generator shell and the vertical baffle that separates the boiler and downcomer regions. Branch 306 represents the region at the lower tubesheet, where the flow changes direction from downward to upward.

The boiler region is separated into two parallel flow paths, representing 90% and 10% of the flow area. The paths are connected by crossflow junctions. Components 310 through 323 represent the 90% region while components 360 through 372 represent the 10% region. The split boiler region model is recommended to simulate phenomena during periods of emergency feedwater injection. This injection enters the boiler around the circumference of the boiler, near the upper tubesheet (junction 854 in the model) and is directed radially inward, into the tube bundle. Because the OTSG employs over 15,000 tubes, the emergency feedwater wets only a small portion of the tubes around the periphery of the tube bundle. As the emergency feedwater falls downward, it encounters the tube support plates (there are 17 in the OTSG) that tend to spread the injection flow further into the tube bundle. The split boiler nodalization represents a compromise modeling scheme for simulating this behavior. An initial 10% bundle penetration is expected, and the crossflow connections to the 90% region allow simulation of the inward spreading.

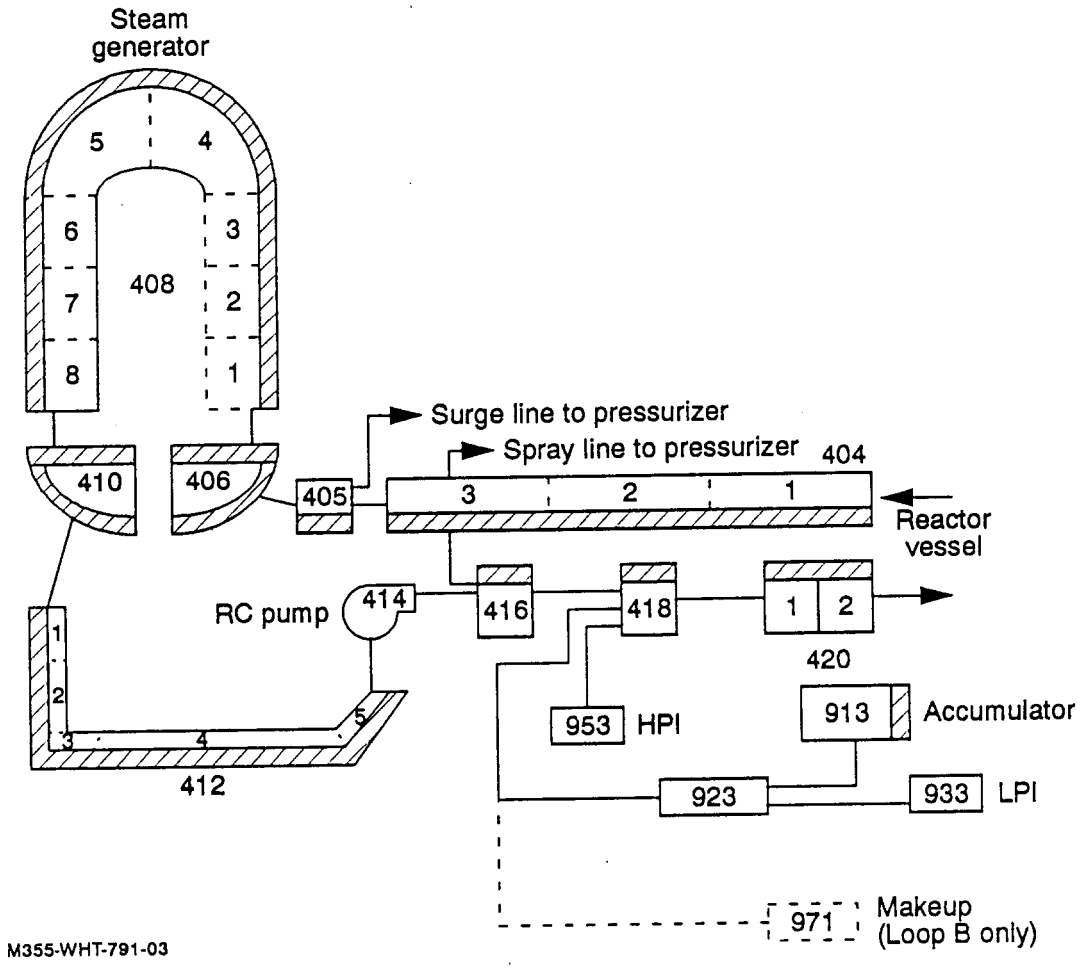

Figure 4-1. Nodalization of primary coolant loops (Loop C shown).

At the top of the boiler region, flows from the parallel boiler channels are combined in branch 325 before exiting the steam generator through a steam annulus, modeled with components 330 and 340.

Modeling the behavior of an OTSG is perhaps the most difficult of nuclear thermal-hydraulic system code problems encountered. The difficulty arises for two reasons. First, a complete spectrum of heat transfer phenomena is experienced between the tube wall and the secondary fluid. At the bottom of the tubes, heat transfer is to subcooled liquid. As the flow progresses up the tubes, the liquid is then saturated and boiled away. To preheat the feedwater, a portion of the steam flow is bled into the downcomer through an aspirator near mid-boiler (modeled with the junction between components 365 and 305 in Figure 4-2). Further up the tubes, any remaining droplets are vaporized and the steam is significantly superheated. Second, the OTSG heat removal rate is very sensitive to the secondary-side liquid level. As the level increases, more of the tube surface area experiences effective heat transfer (e.g., boiling) rather than ineffective heat transfer (e.g., convection to steam). Moreover, the sensitivity of OTSG heat removal to level is present during normal operation, while for UTSGs this is a concern only during accidents that involve an extreme depletion of secondary liquid.

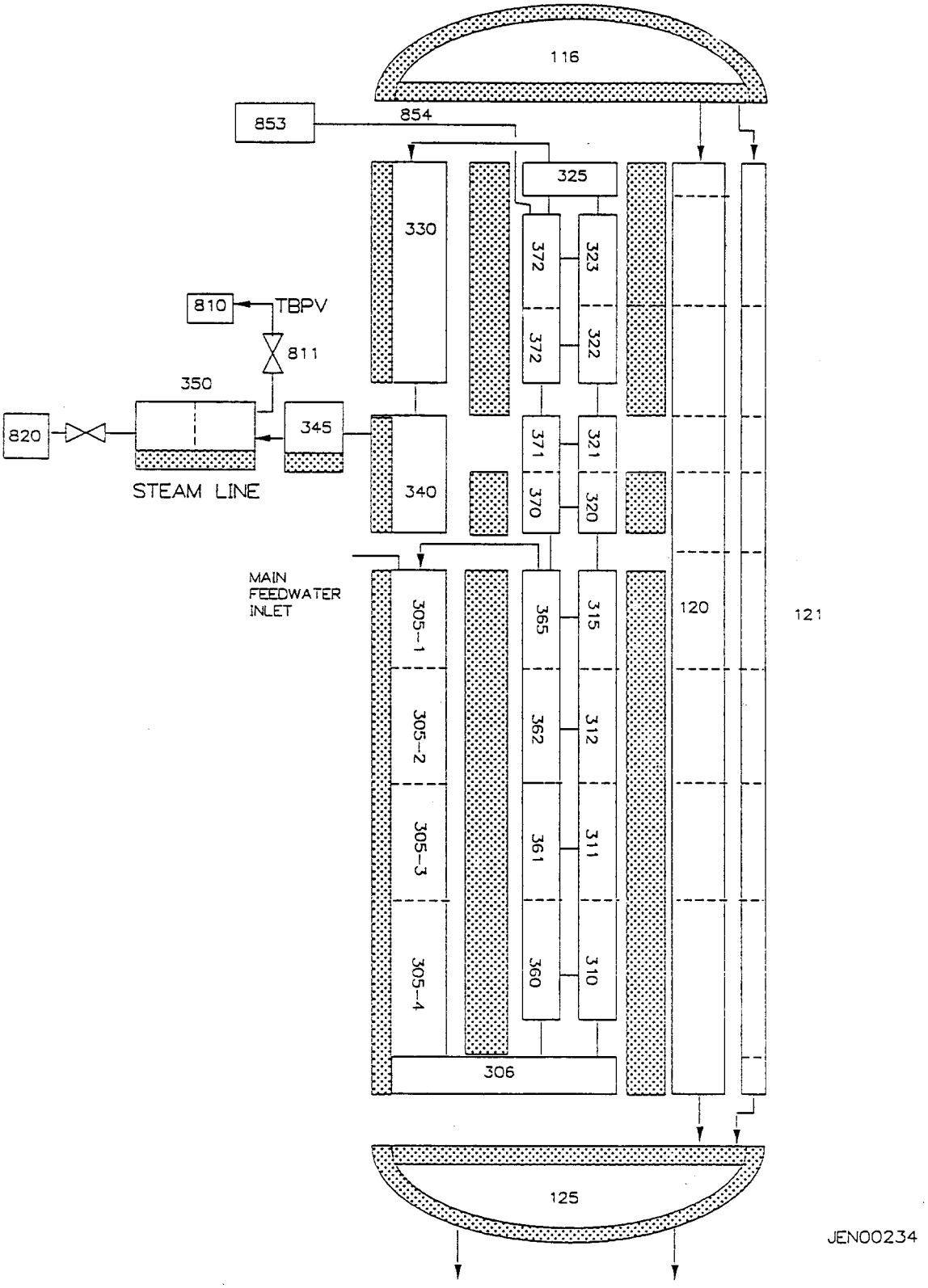

Figure 4-2. Example of Once-Through Steam Generator (OTSG) nodalization.

The OTSG steam generator nodalization shown in Figure 4-2 has proven adequate for simulating normal operation. The difficulty in obtaining a satisfactory OTSG simulation described above is partly nodalization dependent. Nodalization is by nature discrete, and this causes the steam generator heat removal in the model to be even more sensitive to the secondary level than in the prototype. In the model, as the level moves across cell boundaries, discrete jumps in overall heat transfer are encountered. These changes often cause the model to become unstable, oscillating between two solutions at two different secondary levels. Moving to finer axial noding may remedy the oscillation, however the proximity of the liquid level to cell boundaries often is more important than cell size.

4.2.3 Steam Generator Secondaries

Standard INEL nodalization for a U-tube steam generator secondary is shown in Figure 4-3. In the secondary region, main feedwater enters the steam generator downcomer annulus at branch 258 where it is combined with the recirculation liquid flow returning from the separator (component 278) through downcomer annulus branch 254. The combined flow descends through the downcomer (annulus 262) and enters the boiler (pipe 266). Note that the axial nodalization was made consistent between the tube primary, boiler, and downcomer regions. The use of four axial hydrodynamic cells in the boiler region has proven generally useful. However, finer nodalization of the boiler region may be needed for simulating phenomena associated with reflux cooling mode and significantly depleted steam generator secondary inventory. The user is advised to carefully consider the nodalization needs for a particular application. Overall steam generator performance is dependent on correctly simulating the recirculation ratio (the boiler flow rate divided by the feedwater/steam flow rate) because it controls the heat transfer process on the outside of the tubes. The flow losses associated with the horizontal baffles in the tube bundle region often are not well-characterized. Therefore, if a satisfactory initial agreement with the desired recirculation ratio is not attained, adjustment of input form losses in the boiler may be justified.

The two-phase mixture exiting the boiler region flows through the mid-steam generator regions (branches 270 and 274) before entering the separator (branch 278). The separator model is idealized and includes three modes of operation that are determined by the separator void fraction. The void fractions defining these modes are input by the user. At low void fractions, the separator model reverts to a normal branch component, allowing carryover of liquid into the steam dome (branch 282). At high void fractions, the separator also reverts to a normal branch component, allowing carryunder of steam through the liquid return path into the downcomer. At intermediate void fractions, an idealized separation process is calculated: all liquid is returned to the downcomer and all vapor is passed to the steam dome.

The modeler should carefully consider the elevation chosen to locate the separator. In the steam generator model, separation will take place based on the void fraction in the separator volume, whose lower and upper elevations are user-specified. In the actual plant, separation is accomplished in two stages (swirl-vane separators and steam dryers) that reside at two different elevations. Therefore, the model is at best a compromise of the actual separation processes. The selections of separator elevation span and void limits determine when recirculation is interrupted as the secondary mixture levels decline. Note that these levels decline significantly when a steam generator’s heat load is reduced, such as following a reactor trip. The levels also decline significantly during transients where the secondary inventory is depleted, such as during a secondary side LOCA.

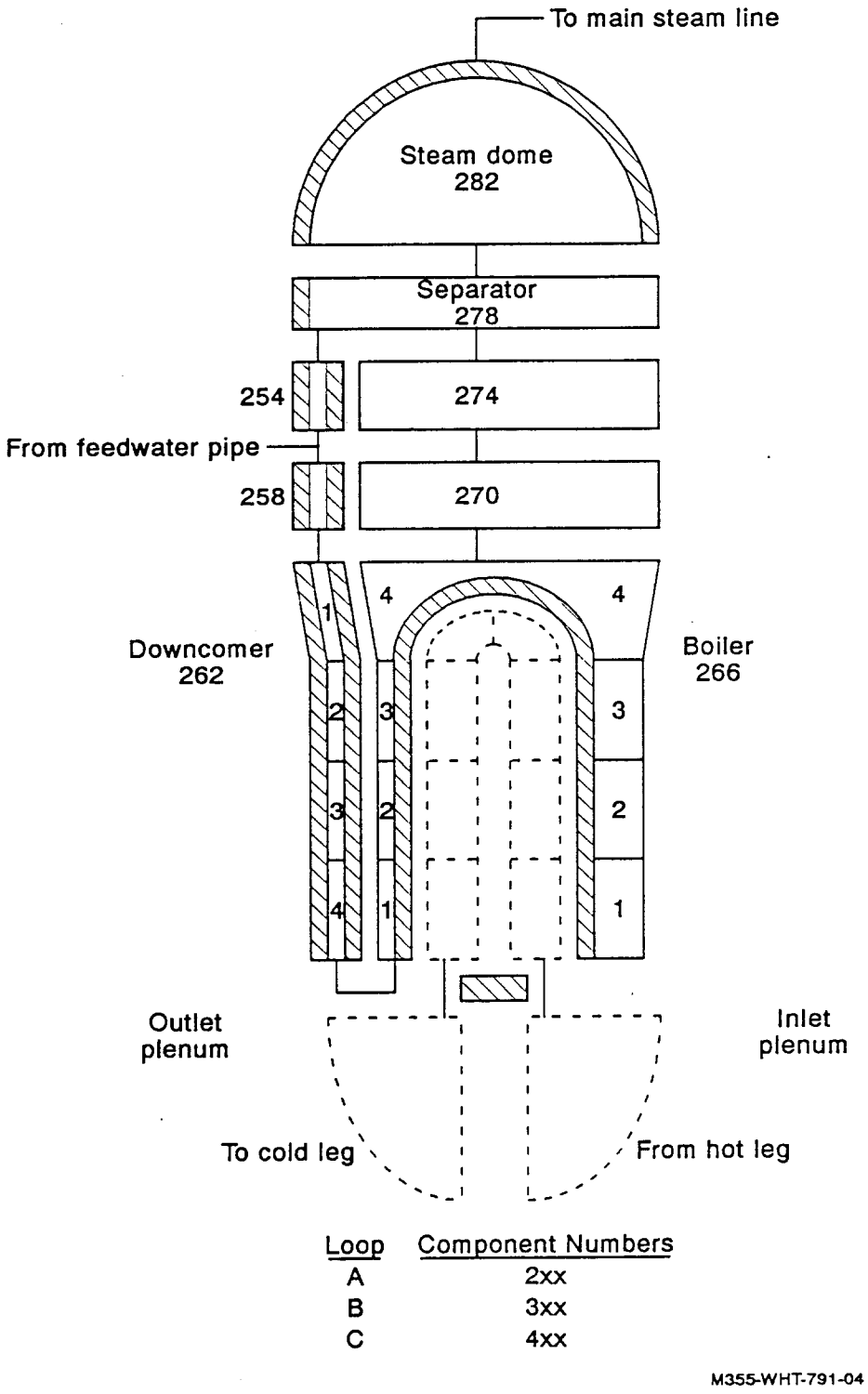

Figure 4-3. Nodalization of secondary side of steam generators.

Heat structures are employed in the model to represent the steam generator tubes, the cylindrical shell and spherical head, the cylindrical baffle separating the boiler and downcomer regions, and the internals of

the separator and steam dome regions.

It often is difficult to obtain a satisfactory agreement with steam generator full-power conditions. The difficulty arises because the heat transfer coefficient calculated on the outside surface of the steam generator tubes is based on general vertical-pipe correlations rather than correlations that account for the swirling flows present within the tube bundle region. The swirling flow pattern results because horizontal baffles in the boiler direct the flow back and forth across the tube bundle instead of allowing the flow to proceed axially (vertically upward) through the boiler. The effect of this discrepancy is that tube heat transfer is understated by the code, resulting in excessively high calculated primary coolant temperatures (the temperatures increase until the core heat is driven across the tubes). Since the source of the calculated error is understood (i.e., a general heat transfer correlation is not appropriate for this application), it is recommended that the modeler "adjust" the heat transfer on the outside of the tubes to remedy the discrepancy.

The recommended adjustment is to reduce the input heated equivalent diameter on the heat structure cards for the outer tube surface. It is recommended that instead of using the boiler region hydraulic diameter as the heated diameter that the minimum tube-to-tube spacing (the distance from the outside of a tube to the outside of its neighbor) be used. If the modeler decides not to follow this recommendation, it will be necessary to compromise an important parameter (such as using a lower secondary pressure, higher primary temperature, or lower feedwater temperature) to simulate full-power steam generator operation.

4.2.4 Primary Coolant Pump

A typical nodalization for the primary coolant pumps is again shown in Figure 4-1. Pipe 412 represents the pump suction cold leg. To ensure proper simulation of behavior in the loop seal region, cell 4 of this pipe is input as horizontal. This orientation allows the formation of horizontally stratified flows at the bottom of the loop seal. It is recommended that at least one horizontal cell be used for simulating loop seal phenomena. Cells 1, 2, 3, and 5 of pipe 412 provide sufficient vertically-oriented calculational cells for simulating the formation of liquid levels in the loop seal region and for simulating countercurrent flow limiting phenomena.

The pump discharge cold leg is modeled with branches 416 and 418 and pipe 420. This nodalization scheme has proven suitable for simulating horizontal stratification of fluid within the cold legs during loss-of-coolant accidents. The nodalization also provides for proper simulation of the fluid temperature distribution in the region; the junction between the branches is located such that the ECC injection site is correctly modeled. The user should remember that SCDAP/RELAP5 provides a one-dimensional representation of the flow and therefore is not capable of resolving thermal stratification of warm and cold liquids within the same pipe. Therefore, although the model may observe the bulk movement of cold ECC liquid toward the core, it is not capable of observing a stream of cold liquid residing in the bottom of the horizontal pipe. The high and low pressure ECC functions are modeled with pairs of time-dependent volumes and junctions. The ECC fluid injection temperature is specified by the time-dependent volume while the injection flow rate is specified as a function of the cold leg pressure by the time-dependent junction. This method allows simulating the head/flow characteristics of the centrifugal ECC pumps. A SCDAP/RELAP5 accumulator component is used to simulate the injection behavior of the nitrogen-charged

accumulators. This lumped-parameter component model mechanistically represents the tank and surge pipe hydrodynamics, heat transfer from tank wall and water surface, water surface vaporization to the gas dome, and gas dome condensation.

4.2.5 Pressurizer

Standard INEL nodalization for the pressurizer and its associated systems is shown in Figure 4-4. The pressurizer upper head is modeled with branch 340 and the pressurizer cylindrical body and lower head are modeled with 7-cell pipe 341. Generally, good agreement with experimental and plant data has been attained for slow and fast pressurizer insurges and outsurges with this nodalization. The surge line is modeled with 3-cell pipe 343.

The functions of the two power-operated relief valves (PORVs) are lumped into valve 344 and those of the three code safety valves are lumped into valve 346. The valves open in response to a significant primary coolant system overpressure. Operation of these valves, including their hysteresis effects, is simulated using the methods described in example 2 in Section 5.4.2 of Reference 2. The pressurizer spray system is modeled with single volumes 335, 337, and 339, and valves 336 and 338. The spray valves open in response to a mild primary coolant system overpressurization. Operation of these valves is simulated using logic similar to the PORV and code safety valves. The flow area of all valves is that necessary for delivering the rated flow capacity at the rated upstream pressure.

Heat structures are used to represent the cylindrical pressurizer shell and its spherical lower and upper heads, and the pressurizer surge line pipe wall. Heat structures are also used to simulate operation of the pressurizer heaters. Heater power is increased in response to an underpressurization of the primary coolant system pressure and is terminated if a low pressurizer level is sensed.

4.3 Break Nodalization

4.3.1 LOCA Modeling

A common code application is simulating a loss-of-coolant accident (LOCA) involving the full or partial rupture of a coolant pipe within an air-filled containment. These applications may involve experimental facility or full-scale plant LOCA simulations.

The need to adequately measure the break flow in an experimental facility usually dictates a complex experimental break geometry to provide clearance for instrumentation. The experimental facility break design often involves a side pipe leading from the broken pipe to a break orifice and valve. This complex design is best modeled in detail (i.e., the geometry upstream and downstream of the break should be modeled directly). Courant limiting considerations will be important in this application because the fluid velocities in the pipe leading to the break will be large. In most analyses of experimental facility LOCAs, benchmarking the break flow path has been necessary to compensate for uncertainties in the break path resistance and the code break flow models. The benchmarking process consists of using experimental data that characterize the break resistance to adjust the model flow losses for an adequate comparison between measured and calculated break flow. The adjustment is typically accomplished by adjusting the discharge coefficients on the

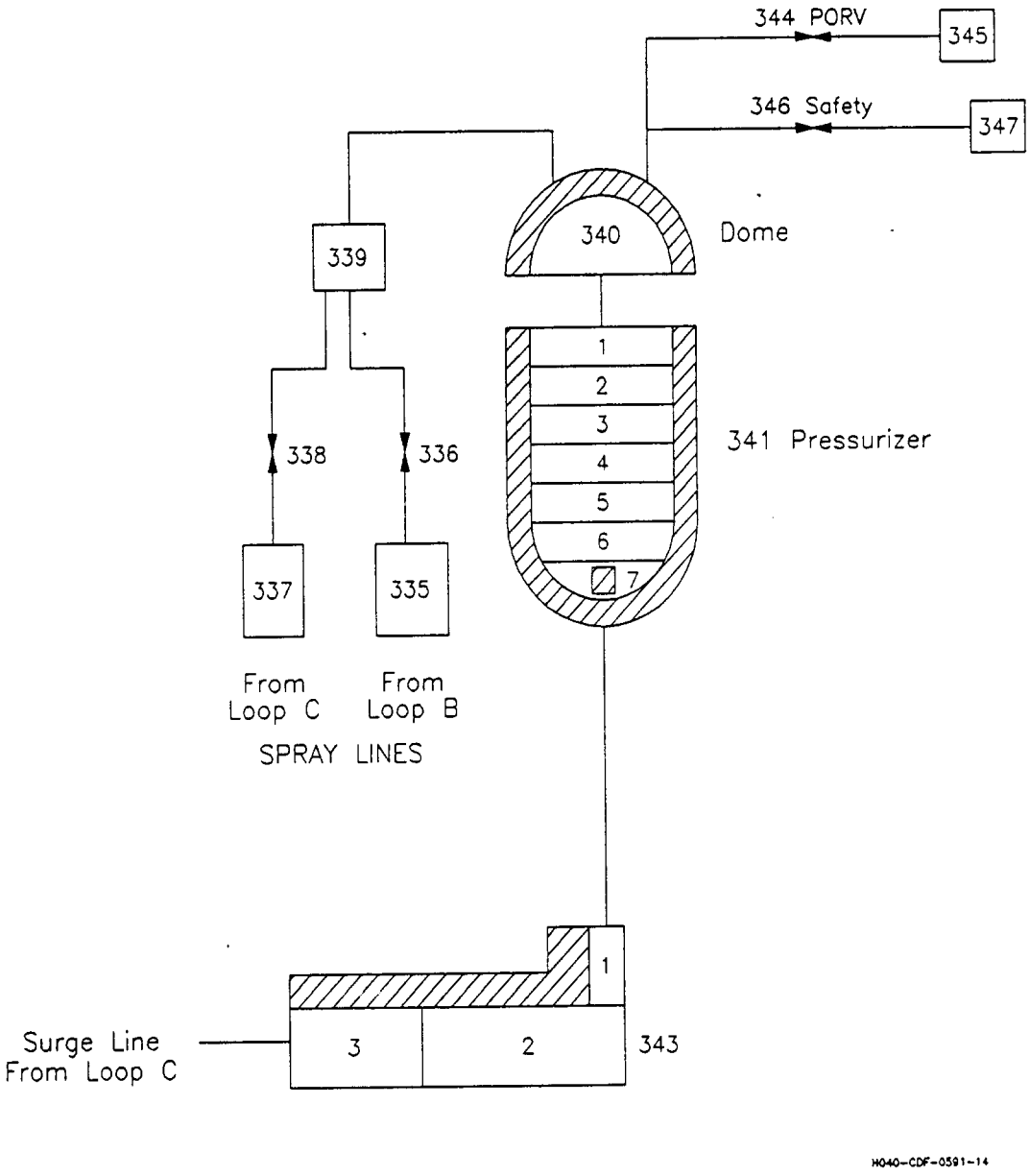

Figure 4-4. Nodalization of pressurizer

break junction.

For full-scale plant applications, the break modeling process typically is more straightforward because the break geometry is simpler. Common LOCA applications for full-scale plants include the opening of circular breaks on the top, side, or bottom of a coolant pipe and the double-ended break of a coolant pipe. For

full-scale plants, breaks typically are assumed to open instantly. Figure 4-5 shows a recommended nodalization for modeling small and double-ended breaks in a coolant pipe. In both applications, the broken pipe is simulated with volumes 110 and 120.

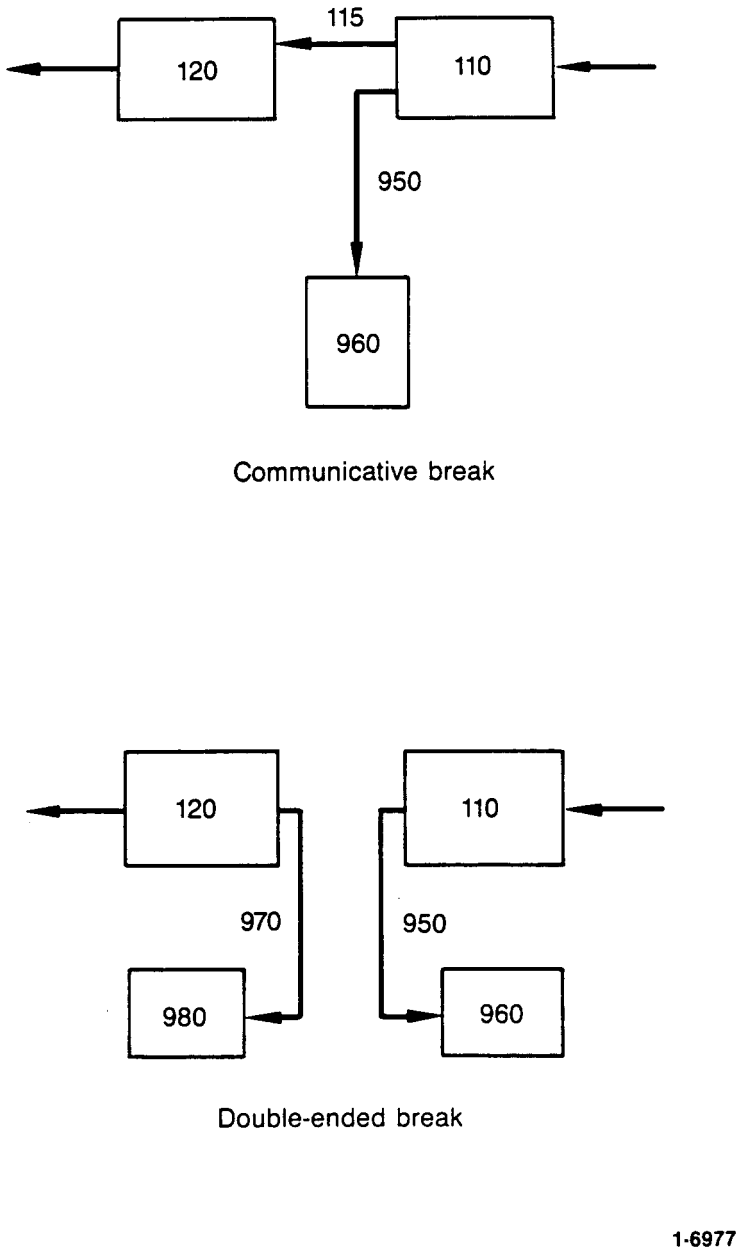

Figure 4-5. Coolant system break modeling.

The small communicative break is simulated by adding single junction 950 and TMDPVOL 960 to the existing hot leg pipe model. The term “communicative” implies a portion of the normal flow through the pipe continues after the break is opened. Note that the break components may be installed on restart, at

the time of break opening, by including components 950 and 960 in the input stream. Break junction 950 should employ the abrupt area change option, simulating the combined flow losses associated with the sharp-edged area reduction from the pipe to the break plane and the sharp-edged expansion from the break plane to the containment. Junction 950 should employ the choking option and be initialized at a zero flow condition. The junction control flags provide the capability to locate the break on the top, side, or bottom of the pipe.

TMDPVOL 960 simulates the containment into which the break discharges; this implies the containment state is a boundary condition in the calculation. Frequently, a constant-pressure containment assumption is used. If the containment pressure response is known (e.g., as a function of the integrated break flow), then that response may be included in the simulation. For the double-ended break the nodalization includes two break junctions and two TMDPVOLs, as shown in Figure 4-5. Note that two TMDPVOLs are needed because no more than one junction may be attached to a TMDPVOL. As for the small break, the break junctions should employ the abrupt area change and choking options. Care should be used when specifying the initial break conditions. In the example shown, the initial mass flow rate for junction 950 should be positive at the same rate as at the inlet to volume 110; the initial mass flow for junction 970 should be negative and of the same magnitude.

In the above examples, the breaks also could have been implemented by including trip valve components at the break junctions in the original model rather than by adding them on restart. The valves would then be tripped open at the time of the break. Using this technique, the breaks may be opened at any time, not just at a restart point.

The containment condition specification is more important in some applications than in others. For small break applications, the primary coolant system depressurization typically is small, the pressure drop across the break remains large, and the break flow remains both choked and positive (into the containment). The containment conditions specified in this situation are not particularly significant to the simulation. The problem is only moderately sensitive to the containment pressure and is insensitive to its gas species. However, for large breaks, transitions between choked- and friction-dominated flow, and intermittent reverse flow from the containment are likely. In this case, it is important to adequately specify the containment conditions.

For some problems where the response of the containment is particularly important, it may be possible for an approximation of the containment behavior to be included as a part of the model. This could be accomplished by modeling the containment and the actual containment mass and heat balances. The code has not been extensively applied in this manner, however.

As a final note, the analyst should appreciate that critical break flow simulation represents an area of significant uncertainty. For some problems, this uncertainty may be a controlling factor for the outcome of the simulation. It is therefore recommended that care be taken to independently check code-calculated break flow results either against experimental data in similar geometries or against standard critical flow correlations.

A recommended procedure for correctly specifying the break area and discharge coefficient is linked to the break scenario, the break plane geometry, and whether any data exists for that geometry. Assuming a

discharge coefficient of 1.0 is valid, the following generalities are known concerning the SCDAP/RELAP5 critical flow model:

- For subcooled conditions, the SCDAP/RELAP5-calculated flow is too large. Often, it is found that a discharge coefficient of about 0.8 is needed to predict break flow in representative geometries containing break nozzles with length-to-diameter ratios less than 1.0.
- For low-quality saturated conditions, SCDAP/RELAP5-calculated mass flow rates are too low, often by as much as 20%, even when a discharge coefficient of 1.0 is used.
- Higher-quality saturated conditions at the break plane, such as are approximated by the homogeneous equilibrium model, are well-simulated with SCDAP/RELAP5.

4.3.2 Surge Line Modelling

One of the transient phenomena which is unique to severe accident analysis is failure of the pressurizer surge line. Surge line break modelling differs from most breaks because of the fact that the timing of the failure is not a boundary condition, but is calculated by the SCDAP/RELAP5 code. Modelling of the failure of a surge line can be performed in one of the following two methods.

METHOD 1:

1 Model the surge line walls with RELAP5 heat structures.

2 Identify surge line heat structures for the creep rupture calculation, on cards 21000110 and 21000000.

3 Specify the ‘DCREPH’ variable on 208 cards to allow it’s use with a logical trip.

4 Specify a logical trip to be driven ‘true’ when any ‘DCREPH’ variables indicates rupture.

5 Model the surge line failure with a valve from the surge line to containment. This valve could be modelled as the communicative break shown in Figure 4-5, with the valve numbered 950. The valve should be initially closed, and open when the trip specified in step 4 is driven ‘true’.

METHOD 2:

1 Model the surge line walls with RELAP5 heat structures.

2 Identify surge line heat structures for the creep rupture calculation on cards 21000110 and 21000000.

3 Perform a calculation to identify the time of the first creep rupture.

4 Restart the calculation with a break being defined for the time of creep rupture.

The flow area of the valve which models the surge line rupture, is probably plant and transient specific. Analysts at the INEL have surveyed a number of creep rupture failure experiments and have concluded that generally the failures showed longitudinal cracks that had opened to various degrees, from about 1/4 of the axial flow area to an area at least as large as the axial flow area. It should be noted that there is a wide scatter of experimental data in this area. In the model for surge line rupture for the Surry plant[15], a rupture flow area equal to 1/3 of the pipe axial flow area was judged appropriate. This amounts to a valve diameter of approximately 1/2 of the surge line pipe.

5. PROBLEM CONTROL

5.1 Problem Types and Options

SCDAP/RELAP5 provides for four problem types: NEW, RESTART, PLOT, and STRIP. The first two are concerned with simulating hydrodynamic systems; NEW starts a simulation from input data describing the entire system; RESTART restarts a previously executed NEW or RESTART problem. PLOT and STRIP are output type runs using the restart plot file written by NEW or RESTART problems. NEW and RESTART problems require an additional option to be selected, STDY-ST, or TRANSNT.

A RESTART problem may restart from any restart record. A note indicating the restart number and record number is printed at the end of the major edit whenever a restart record is written. The restart number is equal to the number of attempted advancements and is the number to be used on Card 103 to identify the desired restart record. The record number is simply a count of the number of restart records written, with the restart record at time equal zero having record number zero. Quantities written in the restart plot records by default are noted in the input data description in Appendix A. User specified input can add additional quantities to the restart plot records in Appendix A.

PLOT and STRIP are output type runs. PLOT generates plots from data stored on the restart plot file. STRIP writes selected information from a restart plot file onto a new file. The new file consists of records containing time and the user selected variables in the order selected by the user. Data to be plotted or stripped are limited to that written in the plot records on the restart plot file.

5.2 Time Step Control

Input data for time step control consist of one or more cards containing a time limit, minimum time step, requested (maximum) time step, control option, minor edit plot/frequency, major edit frequency, and restart frequency. The time limit must increase with increasing card numbers. The information on the first card is used until the problem time exceeds the card limit, then the next card is used, and so on. In restart problems, these cards may remain or may be totally replaced. Cards are skipped if necessary until the problem time at restart is properly positioned with regard to the time limit values.

Several time step control options are available. Transfer of information between the hydrodynamic and heat conduction advancements is explicit, and the advancement routines are coded so that each advancement can use a different time step. Although not now used, each heat structure can also use its own time step. The time step control option is represented by a number between 0 and 15 that can be thought of as a four bit number. Entering zero (no bits set) attempts to advance both the hydrodynamic and heat conduction advancements at the requested time step. However, the hydrodynamic time step will be reduced such that the Courant limit is satisfied. If out of range water property conditions are encountered, the advancement will be retried with reduced time steps. The problem will be terminated if the time step must be reduced beyond the minimum time step. Each time step reduction halves the previously attempted time step. At the beginning of an advancement for a requested time step, a step counter is set to one. Whenever a reduction occurs, the step counter is doubled. When a successful advancement occurs, the step counter is reduced by one. When the step counter is decremented to 0, the problem has been advanced over one requested time step. Doubling of the time step is allowed only when the step counter is even, and the

step counter is halved when the time step is doubled. With no bits set, the time step is doubled whenever possible. At the completion of advancements over a requested time step, the next requested advancement is obtained and may be different from the previous requested time step if data from the next time step control card are used. If necessary, the new requested time step is reduced by halving until the new actual time step is < 1.5 times the last successful time step.

Setting bit one (entering 1, 3, 5, 7, 9, 11, 13, or 15) includes the features described for entering zero and in addition uses the halving and doubling procedures to maintain an estimate (mass error) of hydrodynamic truncation error within program defined limits. If an acceptable error is not reached and the next reduction would lead to a time step below the minimum time step, the advancement is accepted. The first 100 such occurrences are noted in the output.

If the second bit is set (entering 2, 3, 6, 7, 10, 11, 14, or 15), the heat structure time step will be the same as the hydrodynamic time step. The time step control for the hydrodynamics is determined by the status of the first bit as described above, and both the heat conduction and hydrodynamic advancements are repeated when a time step reduction occurs.

If the third bit is set (entering 4, 5, 6, 7, 12, 13, 14, or 15), the heat transfer will use the maximum time step and the hydrodynamics will use the partially implicit hydrodynamic and heat slab coupling. The time step control for hydrodynamics is determined by the status of the first bit, as described above.

If the fourth bit is set (entering 8, 9, 10, 11, 12, 13, 14, or 15), the hydrodynamics will use the nearly implicit hydrodynamic numerical scheme. The time step can be as large as five times the Courant limit for the TRANSNT option and ten times the Courant limit for the STDY-ST option. The time step control for hydrodynamics is determined by the status of the first bit, as described above.

Note that combinations of the effects of setting of the individual bits are achieved by setting bits in combination. For example, entering five (setting bits three and one) results in the combined effects described above for bits three and one. Older versions of SCDAP/RELAP5 would convert 2 to 3 to maintain compatibility. This is no longer done.

Entering zero is not recommended except for special program testing situations. If bit one is set, care must be taken in selection of the requested time step. Individually, the hydrodynamic and heat conduction advancements are stable; the hydrodynamic time step is controlled to ensure stability, the heat conduction solution with constant thermal properties is stable for all time steps, and the change of thermal properties with temperature has not been a problem. The explicit coupling of the hydrodynamic volumes and heat structures through heat structure boundary conditions can be unstable, and excessive truncation error with large time steps can occur. This has been observed in test problems. Entering three usually eliminates the problem, but often with unnecessary calculations. Judicious use of this option during dryout and initial rewetting may be cost-effective. Most LOFT and Semiscale simulations have entered three for the entire problem.

The minor edit, major edit, and restart frequencies are based on the requested time step size. A frequency n means that the action is taken when a period of time equal to n requested time steps has elapsed. The edits and the restart record are written at time zero and at the specified frequencies up to the

time limit on the time step control card. The maximum time step is reduced if needed, and the edits and restart record are forced at the time limit value. Actions at the possibly new specified frequencies begin with the first advancement with a new time step control card. A restart forces a major and minor edit to be written, and a major edit forces a minor edit to be written. Plot information is written to the internal plot and restart plot files whenever a minor edit is written. Note that minor edits are produced only if minor edit requests are entered; a plot file is written only if plot requests are entered; and plot and restart data are written on the restart plot file only if the file is requested.

An option used for program testing can force a plot print, minor edit, major edit, or combinations of these to be written at each advancement. Care should be used, since considerable output can be generated.

Major edits forced by the program testing option or the last major edit of the problem terminated by approach to the job CPU limit may not coincide with the requested time step. When this occurs, a warning message is printed that states that not all quantities are advanced to the same time points.

The control option is a packed word containing a major edit select option, a debug output option, and the time step control. The major edit select option allows sections of major edits for the hydrodynamic volumes and junctions, heat structures, and statistics to be skipped. The debug output option forces any combination of plot, minor edits, or major edit output to be written at each successful advancement rather than at just the completion of advancement over a requested time step. All options can be changed with each time step control card.

5.3 Printed Output

A program version identification is printed at the beginning of printed output and the first page following the listing of input data.

5.3.1 Input Editing

Printed output for a problem begins with a list of card images, one per line, preceded by a sequence number. The sequence number is not the same as the card number on data cards. Notification messages are listed when data card replacement or deletion occurs. Punctuation errors, such as an alphabetic character in numeric fields, multiple signs, periods, etc., are noted by an error message; and a $ is printed under the card image indicating the column position of the error.

Input processing consists of three phases. The first phase simply reads and stores all the input data for a problem such that the data can later be retrieved by card number. Error checking is limited to punctuation checking, and erroneous data flagged during this phase nearly always causes additional diagnostics in later phases. The second phase does the initial processing of data. Input data are moved and expanded into dynamic arrays sized for the problem being solved, and default options are applied. Processing and error checking is local to the data being processed. That is, when processing a single junction component, no checking is performed regarding the existence of connected volumes. Similarly, hydrodynamic volumes connected to heat structure surfaces are not checked during processing of heat structure boundary data. At the end of this phase, all data cards should have been used. Unused cards are considered errors and are listed. Asterisks following the card number indicate that the card number was

bad, an error was noted in the card image listing, and that the number is the sequence number rather than the card number. The third phase completes input processing and performs requested initialization. Once the second phase has been completed, data specifying linkages between various blocks of data can now be processed and checked. Examples of error checking are junction connections made to nonexisting volumes, heat structure surfaces connected to nonexisting hydrodynamic volumes, specified thermal properties, and power data not entered. Solution of steady state heat conduction for initial temperature distribution in heat structures is an example of initialization.

Depending on the type of data, input is edited in only one of the last two edits or in both of them. Error diagnostics can be issued during either phase, even if no editing for the erroneous data is done in a phase. When an error is detected, possible corrective actions are disregarding the data, which usually leads to other diagnostics; inserting benign data; or marking data as being entered but useless for further processing. These actions are taken so that (other than errors on problem type and options) input processing continues despite severe errors. Regardless of errors, all data are given preliminary checking. Severe errors can limit cross-checking. Correcting input errors diagnosed in a submittal may lead to other diagnostics in a subsequent submittal, as elimination of errors allow more detailed checking. Except for exceeding requested computer time and printed output limits, any abnormal termination is considered a programming error and even exceeding computer time limits is prevented during transient execution. The final message of input processing indicates successful input processing or that the problem is being terminated because of input errors.

5.3.2 Major Edits

Major edits are an editing of most of the key quantities being advanced in time. The amount of output depends on the input deck and output options chosen by the code user. Output includes a time step summary, trip information, reactor kinetics information, one to four sections of hydrodynamic volume information, hydrodynamic volume time step control information, one or two sections of hydrodynamic junction information, metal-water reaction information, heat structure/heat transfer information, heat structure temperatures, reflood information, reflood surface temperatures, cladding rupture information, control variable information, and generator pump, turbine, and accumulator information. Major edits are quite lengthy, and care should be used in selecting print frequencies. Some sections of major edits can be bypassed through input data on time step control cards.

5.3.3 Cladding Oxidation and Rupture Information

If the user has activated the metal-water reaction model by using an icccg003 card, the cladding inside and outside oxide penetration depth is printed before the heat structure output. An example is where there are two stacks of eight heat structures. The second stack at Elevation 5 shows some inside cladding oxidation. This is because this elevation has ruptured. The pressure is the pressure inside the gap.

5.3.4 Minor Edits

Minor edits are condensed edits of user specified quantities. The frequency of minor edits is user specified and may be different from the major edit frequency. The selected quantities are held until 50 time values are stored. The minor edit information is then printed, 50 time values on a page, nine of the selected

quantities per page, with time printed in the leftmost column on each page. Minor edits can print selected quantities at frequent intervals using much less paper than major edits. Section 4 of Appendix A indicates how to request minor edits and what the user specified quantities represent.

5.4 Edits of SCDAP Heat Structures

The values of variables that describe the state of SCDAP heat structures are printed at the same times that major edits are performed for the RELAP5 calculations. The printout describes the temperature, deformation, and oxidation of fuel rods and control rods and the fission product release from fuel rods. The state of each SCDAP heat structure is printed in the order of its number identifier. In other words, Component 1 is printed first, then Component 2, and so forth.

5.4.1 Temperature Distribution

The first section of printout shows the temperature distribution of the SCDAP heat structure with a component identification number of 1. The fuel centerline and cladding surface temperatures are printed for each axial node. The temperatures have the units of degrees Kelvin. The elevation of each axial node in units of meters is also printed. The radial temperature distribution is shown at the elevation of the midplane of the SCDAP heat structure, and the temperature at each radial node is printed for the midplane elevation.

5.4.2 Cladding Radius

The next section of printout shows the inner and outer radii of the fuel rod cladding. This printout indicates the extent of cladding ballooning. The inner and outer radii are printed for each axial node. The leftmost radius that is printed applies to the lowest axial node and the rightmost radius applies to the highest axial node.

5.4.3 Cladding Oxidation

The next ten lines of numbers that are printed show the results of calculations of cladding oxidation. The oxidation variables are printed for each axial node, with the lowest axial node printed leftmost and the top axial node printed rightmost. The extent of the cladding oxidation is displayed by the line printing the fraction of cladding oxidation at each node. If the value of the fraction of cladding oxidation is equal to one, then the cladding is entirely a shell of ZrO_2.

5.4.4 Meltdown

The next eleven lines of numbers show the extent of fuel rod liquefaction and meltdown. The extent to which liquefied cladding has dissolved the outer part of fuel pellets is shown by the line printing the inner radius of annulus of dissolved UO_2. If no fuel dissolution has occurred, then the printed value of the inner radius is equal to the outer radius of the fuel pellets. The next line of numbers indicates whether a cohesive debris is located at each axial node. If the value of the indicator is 1 at an axial node, then liquefied material from a higher axial node has slumped down and solidified at the axial node. The solidified material has completely filled in the space between fuel rods and has the configuration of a hardpan. The thickness of the hardpan is shown in the next line of numbers. If the fractional height of

cohesive debris is 1.0, then the hardpan thickness is equal to the height of the axial node. The next several lines of printout show the relocation of fuel and cladding. Unless fuel has slumped below the fuel rod, the sum of the mass of UO_2 solidified at each axial node per rod equals the sum of the mass of UO_2 removed from each axial node per rod. The same rule holds for cladding. If the mass of zirconium removed from an axial node is greater than zero, then all the metallic zirconium has slumped from that node and oxidation no longer occurs at then node.

5.4.5 Fission Product and Aerosol Release

The next several lines of printout show the results of calculations of fission product and aerosol releases. The fission product inventory within the fuel is shown by the printout of matrix of numbers. The leftmost column of numbers applies to the lowest axial node (axial Node 1), and the rightmost applies to the highest axial node. Each row of the matrix shows the mass in units of kilograms per axial node per fuel rod of a certain species of fission product. The first row shows the inventory of xenon, the second row krypton, the third row cesium, the fourth row iodine, and the fifth row is the inventory of tellurium as calculated by the PARAGRASS[4-1] fission gas release model.

The balance of the rows show the inventory of aerosols for which the initial masses are input by the code user and for which the release is calculated by the CORSOR model.[4-2] The sixth row shows the retained mass of zirconium. If no aerosol release of zirconium has been calculated by the CORSOR model, then the mass of zirconium will equal the user input mass of zirconium per axial node. Similarly, the seventh row shows the inventory per axial node per rod for iron, the ninth row ruthenium, the tenth row a special isotope of zirconium, the eleventh row barium, the twelfth row strontium, the thirteenth row tellurium, the fourteenth row silver, the fifteenth row a special isotope of cesium, and the sixteenth row a special isotope of iodine.

The next line of numbers shows the inventory of fission products in the fuel cladding gap. The species are printed in the same order as for the printout of the fuel inventory. The leftmost species is xenon, the second leftmost species is krypton, and so forth. In addition, the mass of helium in the gap is printed as the seventeenth number.

The next line of numbers shows the cumulative release of fission products to the coolant. The mass is units of kilograms per rod is shown for each species. The species are printed in the same order as for the printout of the fuel inventory. In addition, the cumulative release of helium and hydrogen are shown as the seventeenth and eighteenth numbers respectively.

The code user can also obtain cumulative release of fission products to the coolant by subtracting the current inventory from the initial inventory. The difference in initial and current inventories is the amount released to the coolant in the case that the cladding has failed. If the cladding has not failed, then the difference is the amount released to the fuel cladding gap.

5.4.6 Cladding Ballooning and Rupture

The next three lines of numbers show the results of the cladding ballooning model. The first line shows the axial node at which the maximum amount of cladding ballooning is occurring. If the cladding has ruptured, it shows the axial node at which rupture occurred. The next line shows the cladding hoop strain at each axial node. The next line shows the pressure of gases in the fuel cladding gap. If the cladding has ruptured, the gas pressure is equal to the coolant pressure at that location.

5.4.7 Fuel Rod Power

The next three lines of numbers show the fuel rod heat generation rate. The first line shows the total heat generation rate (sum of prompt fission power, fission product decay heat, and actinide product decay heat) in units of W/m at each axial node. The next line of numbers is redundant data that are to be ignored by the code user. The third line shows the axially averaged linear heat generation rate.

The remaining lines of printout for the component are redundant and should be ignored by the code user.

5.5 Edits of Fission Product Transport Results

An edit of fission product transport results is included with major edits. This edit is not produced until fission products are released from SCDAP components unless the user initializes the fission product transport model with nonzero masses. The edit is ordered in numerically increasing order by the hydrodynamic system, then by the volumes within the system. Only hydrodynamic systems having fission product transport are listed.

For each volume, the following information is presented for each quality being tracked. The labels in the left column are the identifiers for the quantities:

Src Fission product source to volume (kg/s).

liq Fission product mass in volume carried by the liquid phase (kg).

vap Fission product mass in the volume in vapor form (kg).

tot Total fission product mass in both vapor and aerosol form in the volume (kg).

nn Fission product in aerosol bin number nn (kg).

For volumes connected to heat structures, the bordering surface is identified and the following quantities are edited for each quantity being tracked.

mc Vapor phase fission product mass condensed on heat transfer surface (kg).

ma Vapor phase fission product mass absorbed in heat transfer surface (kg).

mp Aerosol fission product mass deposited oh heat transfer surface (kg).

5.6 SCDAP/RELAP5 Control Card Requirements

Very general procedures using the control language statements have been developed to simplify SCDAP/RELAP5 execution. Control card sequences used to execute SCDAP/RELAP5 on a CRAY using the UNICOS operating system are documented in Section 23, Appendix A.

5.6.1 Transient Termination

The transient advancement should not abort except for exceeding print line limits.

The user may optionally specify one or two trips to terminate a problem. Normal termination is from one of these trips or the advancement reaching the final time on the last time step control card. Minor and major edits are printed and a restart record is written at termination. Since trips can be redefined and new time step cards can be entered at restart, the problem can be restarted and continued.

Transient termination can also occur based on two tests on the CPU time remaining for the job. One test terminates if the remaining CPU time at the completion of a requested time step is less than an input quantity. The second test is similar, but the comparison is to a second input quantity and is made after every time advancement. The input quantity for the first test is larger than for the second test because the preferred termination is at the completion of a requested time step. In either case, the termination can be restarted.

Failure terminations can occur from several sources, including hydrodynamic solution outside the range of water property subroutines, heat structure temperatures outside of thermal property tables or functions, and attempting to access an omitted pump curve. Attempting to restart at the point of failure or at an earlier time without some change in the problem input will only cause another failure. Problem changes at restart may allow the problem to be successfully restarted. Requested plots are generated after a failure termination.

5.6.2 Problem Changes at Restart

The most common use of the restart option is simply to continue a problem after a normal termination. If the problem terminated because of approaching the CPU time limit, the problem can be restarted with no changes to information obtained from the restart file. If the problem stopped because the advancement time reaching the time end on the last time step card, new time cards must be entered. If the problem was terminated by a trip, the trip causing the termination must be redefined to allow the problem to continue. Thus, the code must provide for some input changes for even a basic restart capability.

The ability to modify the simulated system at restart is a desirable feature. The primary need for this feature is to provide for a transition from a steady state condition to a transient condition. In many cases, simple trips can activate valves that initiate the transient. Where trips are not suitable, the capability to redefine the problem at restart can save effort in manually transcribing quantities from the output of one simulation to the input of another. One example of a problem change between steady state and transient is the use of a liquid filled, time dependent volume in place of the vapor region of a pressurizer during steady state. The time dependent volume provides the pressurizer pressure and supplies or absorbs water from the

primary system as needed. The time dependent volume is replaced by the vapor volumes at initiation of the transient. This technique avoids modeling the control system that maintains liquid level and temperature during steady state calculations when they are not needed in the transient.

Another reason for a problem change capability is to reduce the cost of simulating different courses of action at some point in the transient. An example is a need to determine the different system responses when a safety system continues to operate or fails late in the simulation. One solution is to run two complete problems. An alternative is to run one problem normally and restart that problem at the appropriate time with a problem change for the second case.

The problem change capability could also be used to renodalize a problem for a certain phase of a transient. This has not been necessary or desirable for problems run at the INEL. For this reason, techniques to automate the redistribution of mass, energy, and momentum when the number of volumes changes have not been provided.

The current status of allowed problem changes at restart in SCDAP/RELAP5 are summarized below. In all instances, the problem definition is that obtained from the restart tape unless input data are entered for deletions, modifications, or additions. The problem defined after input changes must meet the same requirements as a new problem.

Time step control can be changed at restart. If time step cards are entered at restart, all previous time step cards are deleted. New cards need only define time step options from the point of restart to the end of the transient.

Minor edit and plot input data cards can be changed at restart. If any of the minor edit cards are entered, all previous cards are deleted. New cards must define all desired minor edit quantities. The plot request data cards are handled in the same manner.

Trip cards can be entered at restart. The user can specify that all previous trips be deleted and can then define new trips. The user can also specify that the previously defined trips remain but that specific trips be deleted, be reset to false, be redefined, or that new trips be added.

Existing hydrodynamic components can be deleted or changed, and new components can be added. An especially useful feature is that the tables in time dependent volumes and junctions can be changed. If a component is changed, all of the cards for the component must be entered.

Control system components can be deleted, changed, or added.

Heat structures, general tables, and material properties can also be deleted, changed, or added. If these are changed, all of the cards for heat structures, general tables, and material properties must be entered.

Reactor kinetics can be added or deleted on restart. A complete set of reactor kinetics data must be input, i.e., individual sections of kinetics data may not be specified as replacement data.

In summary, all modeling features in SCDAP/RELAP5 can be added, deleted, or changed at restart.

6. INSTALLATION

At the INEL, the RELAP5, SCDAP/RELAP5, and Athena programs are maintained in common source files with procedures available to separate the codes. Often, multiple versions of the codes are supported. These installation notes are intended to describe the installation of the SCDAP/RELAP5.

6.1 Transmittal Contents

A typical transmittal consists of a transmittal letter, code manuals, a media containing the code, and a record showing the writing of the media. The usual media is a magnetic tape containing a 'tar' file consisting of several compressed files. (The tar program is a Unix utility that gathers multiple files or directories of files into one file and can later extract all or selected files from the one file.) These installation notes assume such a transmittal. Information concerning other media are contained in the transmittal letter. The type of media and the recording details should have been arranged before code transmittal based on the recording capabilities at the INEL and the reading capabilities at the requesting facility.

6.2 Machine and Operating System Considerations

The SCDAP/RELAP5 computer program should execute on a wide variety of scientific computers with minimal modifications. In particular, the code should execute on all 64-bit computers, that is computers using 64 bits for both floating point and integer arithmetic. Sixty four bit computers include CRAY-1, CRAY/XMP, CRAY/YMP, CRAY-2, and CDC-NOS-VE machines. It should also execute on the multitude of 32-bit computers that range from workstations to supercomputers and that have 32-bit integer arithmetic but provide 64-bit floating point arithmetic through double precision operations. Applicable 32-bit computers include Decstation, HP, IBM mainframe, Ibmrisc, Sun, Apollo, and Vax.

The term code portability should be defined. Here it means that a common source file is maintained for all computers, and through one or two stages of precompiling, the code is made suitable for a particular computer. Some would dispute this definition, maintaining that a code is portable only if the source file is directly applicable to different computers. Some would further maintain that such portability can be achieved if standard Fortran-77 is used.

The first version of SCDAP/RELAP5 was written before Fortran-77 was available. Features of Fortran-77 have eliminated several hardware dependencies. These include use of generic functions, character variables and statements, and open statements. The prior packing and unpacking of small integer and logical quantities was highly machine dependent. Most of the packing and unpacking has been removed, and the remaining packing and unpacking of mostly one bit quantities uses word length independent techniques. The bit handling functions used are from a Mil-spec standard and many computers have implemented that standard. For systems that have not implemented the standard, simple one line statement functions and in some cases an ishft function in the environmental library provides the functions. Many of the machine dependencies are needed to overcome deficiencies in the Fortran standard. For example, the standard does not allow specification of variable range and precision requirements. Unless modified, the Fortran for a 64-bit machine in single precision would use 64 bits while the 32-bit machine would use only 32 bits. Additional statements are needed to indicate double precision

on the 32-bit machine. Some compilers have compiler options for automatically converting to double precision, but that is nonstandard and not available on all systems. Other sources of nonstandard coding is the need of some subroutines to know the smallest and largest floating point numbers, the smallest floating point increment, and the ability to access the radix, fraction, and power part of a floating point number. The next standard should remedy these, but until then, precompilers are used to handle hardware and software differences.

The SCDAP/RELAP5 code is no longer supported for the CDC 60-bit machines such as the Cyber-6600 and Cyber-176. The small memory of those machines and the larger memory requirements of the current versions lead to implementation difficulties. Also these machines are rapidly being phased out of service.

At the INEL, SCDAP/RELAP5 executes on a CRAY/XMP-24 using the Unicos operating system. The program is maintained on that machine using the Update program for source maintenance. Scripts for the installation of SCDAP/RELAP5 on the INEL CRAY machine using the Update program are on the transmittal tape. SCDAP/RELAP5 has also been installed on CDC-NOS-VE machines, IBM mainframe machines, several types of Unix based workstations including Decstations, Vaxstations, Apollo workstations, Sun workstations, IBM-Risc workstations, and even a personal computer. Scripts for a Unix based installation using the make utility are on the transmittal tape. These scripts do not use the Update program and should serve as a base for installation on any Unix machine including the CRAY. Even with the use of the Unix system on many computers, it still is not possible to easily write installation scripts for all Unix machines since there are differences in the Fortran compiler's name and options and the library maintenance program's name and options. The scripts shown here are an attempt at a common script and use "C-shell" if ...endif logic to select machine specific options. Much of the machine dependent commands merely set shell variables, which are passed to the part of the script common to all machines. Options are provided for several machines, and the scripts have been tested on several computers.

The current versions of SCDAP/RELAP5 have been installed on CDC-NOS-VE and IBM mainframe computers but the scripts or control language for installation on the these machines are not available from the INEL. Since the SCDAP/RELAP5 staff does not use these computers, it is difficult to obtain the skill to develop efficient installation scripts. We request that recipients of the code that have developed such procedures to document them for others' use. Hopefully, from the descriptions of the scripts in these notes, a user knowledgeable of his operating system will be able to install the code.

6.3 Transmittal Tape Files

Listed below are the files needed for installation of SCDAP/RELAP5. Additional files may be present on the transmittal tape and additional files would usually be additional sample problems and printed output. The first name is the file name used in the scripts. The scripts assume that the files are transferred to disk using the first file names and that the files are stored in one directory. The second name is the local file name used when writing the transmittal tape on a Cyber machine. The dayfile from the job creating the tape is usually included in the transmittal. In a change from previous transmittals, the file names are now generic, that is they are not coded to indicate the code version, so that generic scripts can be used.

1. selap.s SELAPS SCDAP/RELAP5 source in Update source format.

2.	envrl.s	ENVRLS	Environmental library source in Update source format.
3.	matpro.s	MATPROS	Material property library source in Update source format.
4.	usplit.f	USPLITF	Utility program for separating the individual files packed into a common file.
5.	select.f	SELECTF	Precompiler program to select the appropriate code for a particular computer and operating system.
6.	cnv32.f	CNV32F	Precompiler program to convert code for 32-bit machines.
7.	instls	INSTLS	Master script for installation of RELAP5 using the Update program.
8.	dinstls	DINSTLS	Master script for installation of RELAP5 on Unix systems and using the make utility.
9.	dutilty	DUTILTY	Script to compile and load usplit, select, and cnv32 programs.
10.	goodies	GOODIES	Packed file containing other scripts, makefiles, and text files for installing RELAP5.
11.	indecks	INDECKS	Packed file containing input decks.

The usual transmittal tape format consists of the files above, written in a format agreed upon by the recipient and the INEL. This normally requires some individual processing for each file. An alternate tar format is available for Unix machines. All the files are gathered into one file by the command

tar cv *

where it is assumed that the transmittal files were the only files in the directory. When this file is transferred to the machine for the new installation, the command below retrieves the individual files.

tar xv

Most of the files written in a tar file will usually be compressed and have a .Z suffix. To uncompress the files enter

uncompress *

The compressed files with the .Z suffix will now be uncompressed and will no longer have the .Z suffix.

6.4 Upper and Lower Case Considerations

The magnetic tape is in ASCII with both upper and lower case characters. At the INEL, the Fortran source files are maintained and compiled in lower case. Accordingly, the required alphabetic input (e. g., snglvol specifying a hydrodynamic component type) must be in lower case. User supplied names may be in any combination of lower or upper case. Lower case is used for convenience in maintaining source and input decks since editor commands use lower case extensively. Many format statements, especially those writing sentences, use upper case for standard capitalization.

Most installations have utilities and editors that can switch files between upper and lower case. The Fortran and input decks can be converted to upper case if desired or needed. One C coded subroutine (readnonb) in the environmental library used for CRAY Unicos installations and the scripts should be used as transmitted. If the deck names are capitalized, the list of .F files in the Makefiles should be changed to upper case letters.

6.5 Installation Procedures

Two installation scrips are maintained, one based on the Unix make utility, the other based on the update utility.

6.5.1 Make Based Installation

The next subsection describes loading the transmittal files assuming that the transmittal media is a magnetic tape containing a 'tar' file written by the Unix tar utility. The following subsection states the requirements to be met when unloading the code from a different media or format. The third section describes the installation after the code information is extracted from the media.

6.6.1.1 Tar File Installation. The transmittal tape was written on a workstation using a command similar to:

tar cv selp8_Version

On some workstations, the tar command is used but, in addition, a dd command may actually write the tar file. The command above contains a version indicator. That number will be used in this description but the actual version number indicated in the transmittal letter or on the tape label should be substituted.

To read the tape, execute a command similar to:

tar xv

or

tar cvf tape_name selp8_Version

The second form is used if the desired tape unit is not the defined unit. If a dd command was used to write the tape, a dd command will also be needed to read the tape. The actual command to read the tape will be written on the tape label inside the tape container. The tar command reading the tape will create a selp8_Version directory within the same directory from which the tar command is issued. The selp8_Version directory will contain this README file and three subdirectories, selp8rt, print, and Verrun. The selp8rt subdirectory contains ten or eleven files in compressed format containing the program source, installation scripts, and sample input. There will be only ten files if the SCDAP option is not included in the transmittal.

Assuming no directory change since the tar command to read to tape, transfer to the selp8rt directory by executing the command:

cd selp8rt

The following command uncompresses the code

uncompress

6.6.1.2 Non Tar Format Transmittal. The ten or eleven files containing the code must be unloaded from tape into a directory containing only the code files. The procedures for unloading the file depend on the media and software used to record the files. Often there will be separate processing for each file. The transmittal letter should include enough information to effectively transfer from the media to the directory on disk.

6.6.1.3 Installation From Disk Files. This description assumes that the current working directory is the one containing the ten or eleven files containing the code. The following commands are then executed:

```
chmod 755 !$
ln * !$
cd !$
dinsntls |cray   |  relap  |nonpa| >&dinstls.out &
         |decrisc|         |npa  |
         |hp     |
         |ibmrisc|
         |sun    |
         |sunnew |
         |sgi
```

The mkdir command creates an additional subdirectory, the chmod command sets permissions for the subdirectory, and the ln command links all the files in selp8rt into selp8r. The cd command transfers to the selp8r directory, and the dinstls command installs the code in that directory. This process of using an additional directory simplifies recovery in case of troubles, since a rm -r * command executed in the selp8r directory removes all files created by the ln and dinstls commands without destroying the original files in selp8rt. After correcting the problem, the ln and dinstls commands can be retried.

The second column shows some of the machines on which this transmittal should execute. This version is capable of executing on other machines as well.

If the NPA subroutines are available, specifying NPA will load the NPA subroutines. To specify the source of the NPA libraries, execute the following command before executing the dinstls command.

setenv NPA directory_containing_NPA_libraries

The dinstls script through the use of lower lever scripts builds utilities, environmental library, thermodynamic property files, material property library, SCDAP/RELAP5 libraries, an executable, and run several sample problems that are routinely run as a new version is created.

The scripts do not destroy any transmitted files, but they do move some files to subdirectories. If something goes wrong, the script cannot simply be restarted. If the problem can be fixed, you can remove all files and subdirectories in the directory used for the installation, relink from the other directory, and reissue the dinstls command.

Transmittals have been tested on a Decstation-5000 (decrisc), an HP workstation (hp), an Ibmrisc-6000 (ibmrisc), and a Sun workstation (sunnew). The print subdirectory contains four additional subdirectories: decrisc, hp, ibmrisc, and sunnew. Each subdirectory contains output files in compressed format from the installation on a machine with the same name as the subdirectory. These files can be viewed or printed after uncompressing. The files with suffix .out contain output from the dinstls script. Print files from execution of sample problems have a .p suffix and the strip files have a .st suffix.

The results are slightly different in some of the sample problems. All machines use IEEE floating point arithmetic, but the handling of underflow may be different on the machines (probably because of software differences). The cause of the differences has not been identified. A diagnostic message has been noticed during execution with the IBMRisc version. A comparable message was not obtained on the Decstation and the results appear satisfactory. The warning messages concerning floating point arithmetic on the Sun workstation results from using an option that causes aborts on illegal floating point operations.

6.5.2 Installation Problems

Methods to resolve installation problems will be described in this section. It is generally expected that the resolution of specific problems would be inappropriate here, unless those problems are generic to all users. Those that encounter problems during installation of SCDAP/RELAP5 should first read the README file, which was included with the transmittal. If that fails, the users is encouraged to contact the code developer's at the INEL.

The most frequent difficulty when installing SCDAP/RELAP5 is the failure of the fortran optimizer. The first step most code developer's will take is to repeat the installation with the optimization reduced for the offending routine. Refer to "Recompilation of Specific Routines" on page 9 for additional details.

6.5.2.1 C Language Difficulties

The most prevalent source of problems during the installation of those portions of the code which are written in the C language, is the lack of an ANSI standard compiler. This has been a particular problem for Hewlett Packard workstations, which default to a non-ANSI standard compiler which is insufficient for installation of SCDAP/RELAP5. The resolution for this difficulty is the installation of the freeware GNU C compiler, called 'gcc'. The source for this compiler, along with the prerequisite bison parser generator, is available at a number of sites, and contains installation instructions. After installation of 'gcc', it is necessary to modify the transmittal files to make use of the new compiler, rather than the default. This is done in the following manner.

- Make a working copy of the installation files.
- Within the working copy of the installation directory, make a temporary directory, called 'tmp'.
- Change to the 'tmp' directory (cd tmp).

- Unpack the installation support files (tar xf ../install.tar).
- Edit the installation script ‘denvrl’
- locate that portion of the script which is activated for the machine to code is to be installed on.
- replace the ‘cc’ commands with ‘gcc’, and replace the option ‘-aA’ with ‘-ansi’ if necessary.
- remove the installation support files install.tar (rm ../install.tar).
- Create a new installation support file (tar cf ../install.tar).
- Proceed with the installation.

6.6 Utility Routines

This section describes several utility routines and scripts used in installation and maintenance.

6.6.1 Update Program

Complete documentation for the Update program can be found in CDC and CRAY publications. The following is a brief overview of the Update utility.

The Update program is a source code maintenance program that maintains source files as decks and comdecks. Decks are usually main programs, subroutines, and functions, and comdecks are common sections of code that can be included into a deck being prepared for compilation through ‘*call comdeck’ statements. The decks and comdecks are maintained in a file called an oldpl. An oldpl is created during a creation run by reading an input file where the beginning of comdecks and decks are marked by *comdeck name and *deck name cards. The source files in this transmittal are in the proper format for input to Update. Once the oldpl has been created, modifications to decks and comdecks can be made, new oldpl's can be created containing the modifications, and decks can be extracted for compiling. A useful feature is that if a comdeck (usually containing a common block) is changed, each deck referencing that comdeck can be automatically extracted for compilation.

Another useful feature is that compile time options can be handled by Update. Define variables can be set defined or undefined. These define variables have no relation to the variables in the source code maintained by Update. When a statement *if def,name,n is encountered in a deck or comdeck, the following n lines of code are included in the compile file if name is defined or excluded if name is undefined. If a statement *if -def,name,n is encountered, the minus sign indicates a reversal of the include/exclude logic. If the number of lines is missing, lines up to an *endif card are included or excluded. Nesting of *if def statements is allowed. The CRAY version of Update does not permit the number of lines option, and only *if def and matching *endif statements are used. Define variables are defined by entering *define name statements. The CRAY version also allows define variables to be defined in the control statement.

6.7 Select Program

The action of the Update program in excluding undefined options leads to maintenance difficulties. When looking at a compilation listing of a particular version derived with Update, there is no indication on the listing of the alternate versions. The select program implements logic similar to the Update program using $if [-]def,name[,n] and $endif logic except that excluded cards are not omitted but instead are marked as comment cards by placing an asterisk in column one. Statement numbers are limited to four digits to ensure room for the asterisk. Most compilers recognize an asterisk in column one as a comment card and the select program can be easily modified to use c for the compilers that do not. The $ in the $if def and $endif statements are also changed to * to mark them as comment cards. The result is that options marked with the $ logic are always listed. The define variables for select are defined by $define name statements placed at the first lines presented to select.

Most options are selected using the $ logic. The * logic is usually limited to cases where an entire deck is to be included or omitted.

If the Update program is used, the *if def logic would have been performed by Update and the information passed to the select program would not have *if statements. The Update program also processes *call name statements by replacing in the output file the *call name statement with the text lines headed by *comdeck name. Thus when Update is used, the select program does not encounter *if def or *call name statements. If Update is not used, select handles these statements. The *if def statements are processed in the same manner as $if logic. The *call name statements that Update uses to include comdecks are converted to include 'name.h' statements. Many compilers provide for the include statement to incorporate common coding. The significance of the .h suffix is explained later. Note that when Update is used, the Update program copies the common information stored in the comdecks into the decks. When update is not used, the inclusion of comment text is delayed until the compilation step.

The select program is transmitted in the select.f file.

6.8 Usplit Utility

The usplit utility program is transmitted in the usplit.f file. This program reads a source file suitable for input to the Update program and writes each comdeck beginning with *comdeck name as a separate file name .H. Similarly, each deck beginning with *deck name is written as a separate file name .F. The *comdeck and *deck lines are the first records in the separated files. Text beginning with *hlpdeck name is written as file name and the *hlpdeck is not included in the separate file.

6.9 CNV32 Program

The source for this program is cnv32.f. This program converts the source file, which is oriented to a 64-bit machine to a 32-bit machine version. Changes include: real statements without a length designation are changed to real*8; selected integer and logical variable references with subscripts have 1 or 2 added as the first subscript so that they occupy the same space as real*8 floating point variables; and single precision floating point literals are changed to double precision literals.

The cnv32 program reads a file, mlist, which contains a list of integer and logical global variables that must be converted to occupy the same space as real*8 variables. Local variables needing the conversion are indicated by a list of "*in32 name" statements and terminated "*in32end" in the source. The mlist file needed for processing of the selap file is packed in the file goodies. No mlist file is needed for processing the envrl and matpro files.

6.10 Compilation of Usplit.f, Select.f, and CNV32.f

The usplit, select, and cnv32 programs are written in standard Fortran and have compiled, loaded, and executed successfully on most computers. On the Masscomp computer, the system removed the first column of the output file and to overcome this, an extra blank was introduced at the beginning of each line of output. This change is indicated in comment cards in the source files.

6.11 Recompilation of Specific Routines

There are several scripts which are not used directly in an installation but are used for program modification and debugging and serve to illustrate some features of the installation.

Suffixes are frequently used as part of Unix file names. For example, the usplit program separated comdecks and decks into files with .H appended to comdeck names and .F appended to deck names. Thus comdeck contrl would be in file contrl.H and deck relap5 would be in file relap5.F. The .H and .F are examples of suffixes. Fortran compilers require that the deck to be compiled have a suffix .f and will generate an object deck with .o as the suffix. In this installation, executable files obtained through compilation and loading have a .x suffix.

The doith, doitf, and doitfp scripts are called by

doith comdeckname

doitf deckname

The $1 field of the doith script is a substitutable parameter and is equal to the comdeckname field in the doith command. In the first appearance in the script, the suffix .H is appended and in the second appearance .h is appended. The cat (standard Unix) utility creates a "standard output file" consisting of the define file concatenated to the comdeckname.H file. The define file contain lines of $define name statements and the names appearing in the define statements are set as defined. This means that in $if def,name or *if def,name statements, if the name appears in a $define name statement, name is defined; if the name does not appear in a $define name statement, the name is undefined. In the scripts, the define file is automatically generated based on the machine specified and the options selected. The define file generated for a Decstation for SCDAP/RELAP5 and no NPA linkage is

```
$define decrisc
$define doe
$define erf
$define fourbyt
$define impnon
$define in32
```

```
$define unix
$define selap
$define nonpa
```

The pipe operator, |, indicates that the standard output of the cat command is to be entered directly (piped) as the standard input of select. Thus the input to select is the define file defining the compile time options followed by the comdeck to be processed. Similarly, the standard output of select is piped to cnv32. The standard output of cnv32 is written as comdeckname.h. The resultant file is the comdeck converted for 32-bit machines. These .h files are read by the compiler when processing includes 'comdeckname.h' files generated by the select program from *call statements. The doith script can be viewed as a means of converting .H files to .h files.

The first line of the doitf script is similar to the doith script except that subscripts .F and .f are used. The second line calls the f77 compiler to generate the object file as deckname.o. The doitf script can be viewed as a means of converting .F files to .f and .o files.

The compiler options include generating symbol tables for debugging. Loading the symbol tables for the entire program when debugging can be very time consuming and the initial installation is done without symbol tables. The doitf script is used to compile selected decks with symbol tables as needed for debugging.

6.12 Make Utility

The make program is a Unix utility used to maintain programs. It generates "targets" using an input file describing dependencies between files and rules for generating files from other files. The input files are generally called makefiles and seven makefiles are packed in the goodies file for the Unix type of installation. Except for the list of files to be processed, the makefiles are very similar.

The default target is a library. The dependency statement states that the library file is dependent on the .o files and the rule for creating the library file from the .o files is to use the ar program. The make program checks the creation dates of the target file and the .o files. If one or more .o files do not exist or are newer than the library file, the rule for generating the library must be executed. But the .o files are themselves targets and are dependent on .F and .h files. Similarly, .h files are dependent on .H files. The make program builds a graph of dependencies, and from the creation dates of the files, executes rules as needed to obtain the target file. The .H files are converted to .h files and the .F files are converted to .f and .o files using commands similar to those in the doith and doitf scripts. In an installation, all target files are missing and must be generated. In maintenance applications, an editor is typically used to modify one or more .H and .F files. From the creation dates of the corresponding .H and .h files, only modified .H files will be processed. The modified .F files will be processed but all .F files referring to the modified .H files will also be processed since the dependency rules include information about the *call comdeck statements (or the equivalent include statements after select processing). The .F decks not modified or not having references to modified common decks are not processed.

Included on the transmittal tape is the makefile for the environmental library. This and the other makefiles are ready for installation. If during modification, decks or comdecks are added or deleted, the makefile will need modification. This can be done manually, but most of the changes can be done

automatically. Following the SRCS = \ line is a list of .F files. The user should update that list by adding or deleting decks (with .F appended). No characters, not even blanks may follow the \ which indicates continuation to the next line. No other changes are necessary. Execution of make -f makefilee Deps will update the dependency lists. The list of .o files is directly obtained from the .F list. By searching the .F files for *call statements, the list of .H and .h files and the dependencies between .o and .h files are determined.

To repeat, this step defining the dependencies is not needed for installation. This step has been done at the INEL, and the makefiles as transmitted are ready for installation. This step is needed only if decks or comdecks are added or if decks have *call name statements added or deleted.

6.13 Define Options Used in SCDAP/RELAP5

The following define options are used in the SCDAP/RELAP5 source, the environmental source, or the material property source. The define options selected can be used in processing all three source files. Following the table, each option is described.

apollo	athena	blkdtabufr	cb205	cdc	cdc60
cdccra	cos	cray	ctime	ctss	decrisc
doe	erf	fourbyte	hp	ibm	ibmrisc
ieee	impnon	in32	lcm	mass	mmfld
nopen	npa	plots	selap	sun	timed
unicos	unix	vax	blkdta		

Data statements that load data into common blocks have been gathered into routines that are either block data routines or subroutines containing only a return statement as an active statement. If the blkdta option is undefined, the routines are subroutines and one call to each subroutine is made. The subroutine when called simply returns. The advantage of this over block data is that the subroutine reference will force loading of the subroutine from a library while block data routines are more difficult to load from a library. Loading data into common blocks from a subroutine is not standard but most compilers allow it. Leaving the blkdta option undefined is recommended but defining blkdta can be done if the compiler requires use of block data routines. Some special considerations to load the block data object decks may be needed.

bufr- If the bufr option is defined, buffer in and buffer out statements are used to read and write the rstplt file and the unformatted form of the strip file. The software must also support a record concept and the length function that returns the size of a record read. The bufr defined option is recommended for CRAY-1, CRAY-XMP, and CDC-NOS-VE machines. It is not recommended for the CRAY-2 machine. When the bufr option is undefined, unformatted read and write statements are used. These are standard Fortran statements that can be used with any machine, but may be slower than the buffer in and buffer out statements. In the typical read or write statement, write (rstplt) len, a(i),i=1,len), the implied do loop was not treated as a block write.

cdc - This option is defined for CDC-NOS-VE machines. It no longer signifies CDC-6600 or

7600 machines since those are no longer supported.

cdccra - This option is defined for CDC-NOS-VE machines or CRAY machines.

cos - This option is defined for the COS operating system on CRAY machines.

cray - This option is defined for CRAY machines.

ctss - This option is defined when using the CTSS operating system on CRAY computers.

fourbyt - When the bufr option is undefined, this option selects whether the reads and writes of restart information uses eight or four byte words. This option should be undefined when bufr is defined or when bufr is undefined but the machine uses 64 bits for both integer and floating point data. Thus CRAY and CDC-NOS-VE should have fourbyt undefined. The RELAP5 portion of the code writes restart records that are multiples of eight byte words and the fourbyt option may be undefined on many 32-bit machines. The RELAP5 code executed on IBM Decstation, and Vax machines with fourbyt undefined. The SCDAP portion does require the fourbyt option on 32-bit machines. The safest option for 32-bit machines is to define fourbyt. Compilers should treat the reads and writes as block operations and there should be no difference in execution time.

hp - This option is defined for HP machines.

ibm - This option is defined for IBM mainframe machines.

ibmrisc - This option is defined for Ibmrisc machines.

impnon - Many compilers allow an implicit none statement. This statement when entered in a routine removes all implicit typing of variables and requires each variable and function to be typed. Multiple explicit typing is also considered an error. Additionally, the Cray compiler requires subroutine names called in a routine to appear in external statements. This provides an automatic level of checking by the compiler to avoid inadvertent multiple use of variable names during code development. All comdecks defining common blocks or dynamic blocks have explicit typing. Some subroutines have been modified such that all local variables are explicitly typed. When impnon is defined, an implicit none statement is activated in those subroutines. Eventually, all routines will be explicitly typed. Although most important for code developers, impnon is recommended to be defined for all systems allowing implicit none statements. The IBM mainframe compiler does not allow implicit none and impnon should be undefined for that machine.

in32 - This option is defined for 32-bit machines, that is, machines that can perform 64-bit floating point computations but only 32-bit integer operations. This option is needed for Decstation, HP, IBM, Ibmrisc, Masscomp, Macintosh, and Vax computers.

lcm - This option once referred to the use of LCM storage on Cyber-176 computers. Those computers are no longer supported, and this option should be undefined. It has been left in the environmental library in case support of multiple levels of memory is needed in the future.

mass - This option should be defined for the Masscomp computer.

npa - This option should be defined when linkage to the Nuclear Plant Analyzer (NPA) is

desired. This option should be undefined unless the NPA system has been installed.

plots - This option when defined enables the plot capability contained within SCDAP/RELAP5. The internal plot capability has not been made compatible, is not currently operational on any computer, and should be disabled by having this option undefined.

selap - This option selects between the RELAP5 code and the SCDAP/RELAP5 code. This option when undefined selects the RELAP5 code and when defined selects the SCDAP/RELAP5 code. Some RELAP5 only transmittals will have all SCDAP subroutines removed and some will also have all interface coding removed. For SCDAP/RELAP5 transmittals, this option when undefined selects only the RELAP5 capability, and when defined selects the integrated SCDAP/RELAP5 capability. The SCDAP/RELAP5 code when given RELAP5 only input data generates the same results as if the RELAP5 only code was installed.

unicos - This option is defined for the CRAY computers operating under Unicos.

vax - This option is defined for Vax computers.

6.14 Installation Scripts

The scripts are highly repetitive and consist mostly of if, elseif, and endif statements, which are based on the input arguments define symbols. The symbols primarily define compiler names and options which are not standard on various Unix machines. The common part of the scripts execute commands making use of the symbols to handle system differences.

7. REFERENCES

1. T. Heames et al., *VICTORIA: A Mechanistic Model of Radionuclide Behavior in the Reactor Coolant System Under Sever Accident Conditions*, NUREG/CR-5545, SAND90-0756, Rev. 1, December 1992.
2. C. M. Allison, C. S. Miller, and N. L. Wade (Eds.) *RELAP5/MOD3 Code Manual, Volumes I through V,* NUREG/CR-5535, EGG-2596, DRAFT, June 1990.
3. C. M. Allison and G. H. Beers, "*Comparisons of the SCDAP Computer Code with Bundle Data Under Severe Accident Conditions*," Seventh International SMIRT Conference, Chicago, IL, August 22-26, 1983.
4. E. C. Lemmon, *COUPLE/FLUID A Two-Dimensional Finite Element Thermal Conduction and Advection Code,* EGG-ISD-SCD-80-1, February 1980.
5. J. Rest and S. A. Zawadzki, "*FASTGRASS-VFP/PARAGRASS-VFP Version 50531, Users Guide,*" Argonne National Laboratory Quarterly Report, January-March 1983, Volume I, NUREG/CR-3689, ANL-83-85 Volume I, June 1983.
6. *PATRAN Plus User's Manual*, Release 2.4, PDA Engineering, Costa Mesa, California 1987.
7. *ABAQUS User's Manual*, Version 4.6, Hibbitt, Karlsson & Sorensen, Inc., Providence, Rhode Island, 1987.
8. D. M. Snider, K. L. Wagner, W. Grush, *Nuclear Plant Analyzer (NPA) Reference Manual Mod1*, EGG-EAST-9096, April 1990.
9. K. D. Bergeron et al., *User's Manual for CONTAIN 1.0, A Computer Code for Severe Nuclear Reactor Accident Containment Analysis*, NUREG/CR-4085, SAND84-1204, May 1985.
10. L. T. Ritchie et al., *CRAC2 Model Description*, NUREG/CR-2552, SAND82-0342, March 1984.
11. D. I. Chanin et al., *MELCOR Accident Consequence Code System (MACCS Version 1.5)*, NUREG/CR-4691, SAND86-1562, July 1988, DRAFT.
12. L. J. Siefken et al., FRAP-T6: *A Computer Code for the Transient Analysis of Oxide Fuel Rods*, EGG-CDAP-5410, April 1981.
13. G. A. Berna et al., *FRAPCON-2 Developmental Assessment,* NUREG/CR-1949, PNL-3849, July 1981.
14. J. L. Remple et al., *Light Water Reactor Lower Head Failure Analysis*, NUREG/CR-5642, EGG-2618, October 1993.
15. P. D. Bayless, *Analyses of Natural Circulation During a Surry Station Blackout Using SCDAP/RELAP5,* NUREG/CR-5124 EGG-2547, October 1988.